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

D6.3

Analysis of New Services
Final

Version: 1.0
Author: Janez Povh, ULFME
Date: 20.10.2017

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: < D6.3>	
	Deliverable Nature: < Report >	
	Dissemination Level: PU*	Contractual Date of Delivery: 31 / 10 / 2017
		Actual Date of Delivery: 31 / 10 / 2017
	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: Analysis of New Services	
	ID: D6.3	
	Version: <1.0>	Status: *Final*
	Available at: http://www.prace-project.eu	
	Software Tool: Microsoft Word 2013	
	File(s): D6.3.docx	
Authorship	Written by:	Janez Povh, ULFME
	Contributors:	Abdulrahman Azab (UiO), Martin Bidner (HLRS), Simone Bna (CINECA), Miroslaw Kupczyk (PSNC), Martial Mancip (CEA), Frederic Suter (IN2P3 – CNRS), Xuan Wang (HLRS), Oscar Yerpes (BSC), Riccardo Zanella (CINECA)
	Reviewed by:	Florian Berberich, JUELICH Enver Ozdemir, UHEM
	Approved by:	MB/TB

Document Status Sheet

Version	Date	Status	Comments
0.1	18/9/2017	Draft	
0.3	23/9/2017	Draft	Sent back to serv. 5, 6
0.4	5/10/2017	Draft	proofread
0.5	6/10/2017	Draft	sent for internal review
1.0	20/10/2017	Final version	Submitted to MB

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

List of Figures

List of Tables

References and Applicable Documents

[1] A. Azab, Enabling Docker Containers for High-Performance and Many-Task Computing, in 2017 IEEE International Conference on Cloud Engineering (IC2E). IEEE Computer Society, p 279 – 285

[2] D. Merkel, Docker: Lightweight linux containers for consistent development and deployment, Linux J., vol. 2014, no. 239, March 2014

[3] D. M. Jacobsen, R. S. Canon, Contain this, unleashing Docker for HPC, Proceedings of the Cray User Group (2015)

[4] G. M. Kurtzer, Singularity 2.1.2 - Linux application and environment containers for science, Aug. 2016. [Online]. Available at: https://doi.org/10.5281/zenodo.60736

[5] M. A. Jette, A. B. Yoo, M. Grondona, SLURM: Simple Linux Utility for Resource Management. In Lecture Notes in Computer Science: Proceedings of Job Scheduling Strategies for Parallel Processing (JSSPP) 2003 (2002), pp. 44-60

[6] B. Langmead, C. Trapnell, M. Pop, S. Salzberg, "Ultrafast and memory-efficient alignment of short dna sequences to the human genome", Genome Biology, vol. 10, no. 3, pp. R25, 2009.

[7] PMAP: Parallel sequence mapping tool, [online] Available: http://bmi.osu.edu/hpc/software/pmap/pmap.html

[8] P. Giannozzi et al., QUANTUM ESPRESSO: a modular and open-source software project for quantum simulations of materials, Journal of physics: Condensed matter 21.39 (2009): 395502.

[9] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE, vol.86, no.11, pp.2278-2324, Nov 1998.

[10] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 1097–1105. Curran Associates, Inc., 2012.

[11] J. Povh et al., Analysis of New Services, PRACE 4IP deliverable D6.3, 2015

[12] J. Povh et al., Deployment of Prototypal New Services, PRACE 4IP deliverable D6.4, 2016

[13] M. Kupczyk, D. Kaliszan, H. Stoffers, N. Wilson, F. Moll, Urgent Computing service in the PRACE Research Infrastructure. Available at https://zenodo.org/record/832029

[14] Result of the second Periodic Review of your H2020 project 653838 —PRACE4IP.

[15] http://yann.lecun.com/exdb/mnist/

[16] http://caffe.berkeleyvision.org

[17] https://github.com/intel/caffe

[18] http://www.uio.no/english/services/it/research/hpc/abel/

[19] https://docs.docker.com/terms/layer/
[20] http://environmentalomics.org/bio-linux/
[21] https://github.com/abdulrahmanazab/docker-on-htcondor
[22] https://github.com/soumith/convnet-benchmarks
[23] https://docs.gitlab.com/ee/README.html
[24] https://git-scm.com/documentation
[25] https://trac-hacks.org/wiki/TracGuide
[26] http://www.redmine.org/projects/redmine/wiki/Guide#User-guide
[27] https://jenkins.io/doc/pipeline/tour/getting-started/
[28] https://jenkins.io/doc/book/using/
[29] https://www.eudat.eu/services/b2share
[30] https://skjema.uio.no/prace-containers
[31] http://www.prace-project.eu
[32] http://www.uio.no/english/services/it/research/hpc/abel/help/software/singularity.html
[33] https://depot.galaxyproject.org/singularity/
[34] http://book.mso4sc.cemosis.fr/deliverables/d3.1/
[35] https://github.com/MSO4SC/Singularity/tree/master/examples
[36] https://galaxyproject.org/
[37] https://depot.galaxyproject.org/singularity/
[38] https://lifeportal.uio.no/
[39] M. Ellert et al., Advanced Resource Connector middleware for lightweight computational Grids, Future Generation Computer Systems 23, 219-240, 2007
[40] https://github.com/cea-hpc/pcocc
[41] P. Giannozzi et al., J. Phys.:Condens. Matter 21 395502, 2009, http://www.quantum-espresso.org
[42] https://spark.apache.org.
[43] https://www.tensorflow.org.
[44] http://torch.ch.
[45] http://deeplearning.net/software/theano.
[46] http://neon.nervanasys.com/index.html.
[47] http://hibd.cse.ohio-state.edu.
[48] https://www.ibm.com/support/knowledgecenter/en/STXKQY_4.2.0/com.ibm.spectrum.scale.v4r2.adv.doc/bl1adv_hadoop.htm.
[49] https://github.com/intel.
[50] https://www.ibm.com/ms-en/marketplace/deep-learning-platform.
[51] https://openpowerfoundation.org/.
[52] http://www.grpc.io.
[53] https://github.com/01org/MLSL.
[54] https://arxiv.org/abs/1708.02188?cm_mc_uid=30902614853514913139697.
[55] http://mpi-forum.org.

[56] https://spark.apache.org/docs/latest/.
[57] https://github.com/SparkTC/spark-bench.
[58] https://github.com/intel-hadoop/HiBench.
[59] http://prof.ict.ac.cn/.
[60] https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/General%20Parallel%20File%20System%20(GPFS)/page/2nd%20generation%20HDFS%20Transparency%20Protocol.
[61] http://hibd.cse.ohio-state.edu/static/media/rdma-spark/rdma-spark-0.9.4-userguide.pdf.
[62] http://torch.ch/docs/getting-started.html.
[63] https://github.com/torch/torch7/wiki/Cheatsheet.
[64] [https://software.intel.com/en-us/articles/what-is-intel-daal.
[65] https://www.apache.org/licenses/LICENSE-2.0.
[66] https://software.intel.com/en-us/intel-daal-support/documentation.
[67] https://www.ibm.com/us-en/marketplace/deep-learning-platform.
[68] https://github.com/intel/caffe
[69] https://caffe2.ai/.
[70] https://mila.umontreal.ca.
[71] http://deeplearning.net/tutorial.
[72] https://github.com/Lasagne/Lasagne.
[73] https://keras.io.
[74] https://github.com/intel/Theano.
[75] https://www.tensorflow.org/get_started
[76] https://www.tensorflow.org/api_docs
[77] https://www.tensorflow.org/performance/benchmarks
[78] https://github.com/fchollet/keras-resources
[79] https://github.com/maxpumperla/elephas

List of Acronyms and Abbreviations

aisbl	Association International Sans But Lucratif (legal form of the PRACE-RI)
API	Application Programming Interface
ARC	Advanced Resource Connector
BCO	Benchmark Code Owner
BoD	Board of Directors
BVLC	Berkeley Vision and Learning Center
CAS	Central Authentication Service
CERN	European Organisation for Nuclear Research
CFD	Computational Fluid Dynamics
CoE	Center of Excellence
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture (NVIDIA)
DARPA	Defense 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
ESRF	European Synchrotron Radiation Facility
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.
Geant4	GEometry ANd Tracking
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
GPFS	General Parallel File System
GPU	Graphic Processing Unit
GUI	Graphical User Interface
HA	High Availability
HDF5	Hierarchical Data Format 5
HDFS	Hadoop Distributed File System
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
HTCondor	High Throughput Condor
HTML	Hypertext Markup Language
I/O	Input/Output
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
LHC	Large Hadron Collider
LINPACK	Software library for Linear Algebra
LRMS	Local Resource Management System
LSF	Load Sharing Facility
LSSI	Large Scale Scientific Instrument
LSTM	Long Short Term Memory
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
MSO4SC	Mathematical Modelling, Simulation, and Optimization for Societal Changes with Scientific Computing
NDA	Non-Disclosure Agreement. Typically signed between vendors and customers working together on products prior to their general availability or announcement.
NetCDF	Network Common Data Form
PA	Preparatory Access (to PRACE resources)
PATC	PRACE Advanced Training Centres
PBS	Portable Batch System
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
RDMA	Remote Direct Memory Access
RHEL	Red Hat Enterprise Linux
RI	Research Infrastructure
RNN	Recurrent Neural Network
RoCE	RDMA over Converged Ethernet
SAIO	Semi-Automatically I/O-Tuning Framework
SDK	Software Development Kit
SLURM	Simple Linux Utility for Resource Management
SSH	Secure Shell
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
TileViz	Tile Visualization
URL	Uniform Resource Locator
UNICORE	Uniform Interface to Computing Resources. Grid software for seamless access to distributed resources.
UC	Urgent Computing
VM	Virtual Machine
VMD	Visual Molecular Dynamics
VNC	Virtual Network Computing
VTK	Visualization Toolkit
WLCG	Worldwide LHC Computing Grid

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
CC-IN2P3	Computing Centre of the National Institute for Nuclear Physics and Particle Physics.
CCSAS	Computing Centre of the Slovak Academy of Sciences, Slovakia
CEA	Commissariat à l'Energie Atomique et aux Energies Alternatives, France (3rd 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 (3rd Party to GENCI)
CNRS	Centre National de la Recherche Scientifique, France (3rd 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 développement et des ressources en informatique scientifique, France (operated by CNRS, 3rd Party to GENCI)

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-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

This document presents the work done within Task 6.2 by middle of September 2017. This task aims to (i) continue development of the four services started already in PRACE-4IP within Task 6.2 and (ii) to start development of two new services.

More precisely, we present how to continue the following services initiated in PRACE-4IP:

Service 1: The provision of urgent computing
Service 2: Links to large-scale scientific instruments
Service 3: Smart post processing tools including in situ visualisation
Service 4: Provision of repositories for European open source scientific libraries and applications

Additionally, we consider two new services:

Service 5: Evaluation of lightweight virtualisation technology
Service 6: Evaluation of new prototypes for Data Analytics services

The work in Task 6.2 is coordinated by ULFME. Partners working on this task are grouped in six subgroups, one for each service.

In this deliverable, we present motivations for the continuation of services from PRACE-4IP or for start of new state of the art services, the work done in period January 2017 – midd of September 2017 and the operational plan for the work to be done by the end of the project. We demonstrate that:

- Service 1 needs strategic discussion on PRACE BoD (Board of Directors) levels on how to attract real end-users. The working groups engaged in this service in PRACE-4IP has not detected any real user that would be willing to enter into joint pilot.
- Service 2 is demonstrating very positive developments due to engagement of PRACE BoD (Board of Directors) members, especially the pilots defined by European Synchrotron Radiation Facility (ESRF) which are very promising;
- Service 3 is very relevant and address a large HPC community. The solutions that were pointed out as most relevant will be further tested and deployed on more HPC sites.
- Service 4 has matured enough to move into a regular service of WP6. The transition will be completed by the end of 2017.
- Services 5 and 6 are new and are aligned with the huge importance of adapting container technology for HPC and for convergence between HPC and big data. The two services are also aligned with efforts in WP4 where the MOOC on HPC and Big data has been developed and run and using big data containers will be a precious feature of future runs.

1 Introduction

Important goal of Task 6.2 of WP6 within PRACE-5IP is to develop and test of new services that might be later, after extensive discussion, included into the list of regular services.

Work in this task was planned as continuation of the work conducted in Task 6.2 of PRACE-4IP, which is described in Deliverable D6.4 [12]. We therefore focus on continuation of the four services from PRACE-4IP and on two new services introduced in PRACE-5IP:

1. From PRACE-4IP:
    a. Service 1: The provision of urgent computing
    b. Service 2: Links to large-scale scientific instruments
    c. Service 3: Smart post processing tools including in situ visualisation
    d. Service 4: Provision of repositories for European open source scientific libraries and applications
2. New services in PRACE-5IP:
    e. Service 5: Evaluation of lightweight virtualisation technology
    f. Service 6: Evaluation of new prototypes for Data Analytics services

There are 12 partners having together 78,5 PM in task 6.2. They were divided based on their interest into 6 groups, one for each service. We also defined for each service the service coordinator. Finally we agreed on F2F meeting in Poznan (18-19 April 2017) and teleconferences in May, June and September 2017 on the following distribution of work:

a. Coordinator of Task 6.2: Janez Povh, ULFME;
b. Service 1: Coordinator: M. Kupczyk (PSNC), collaborating partner: NSCA;
c. Service 2: Coordinator: Frederic Suter (CNRS/CC-IN2P3), collaborating partners: CNRS/CC-IN2P3, CNRS-IDRIS, NSCA, CASTORC;
d. Service 3: Coordinator: Simone Bna (CINECA): collaborating partners: HLRS, CEA;
e. Service 4: Coordinator: Oscar Yerpes (BSC): collaborating partners: GRNET, CESGA and NIIF;
f. Service 5: Coordinator: Abdulrahman Azab (UiO): collaborating partners: CEA/DAM, EPCC, CINECA, CASTORC, CNRS-IDRIS;
g. Service 6: Coordinator: Riccardo Zanella (CINECA), collaborating partners: KTH, CNRS/CC-IN2P3, CNRS-IDRIS, EPCC, PSNC.

In the first period of PRACE-5IP (January-September 2017) we have done:

a. Services 1–4: we prepared operational plans for the continuation of work on these four services and made first steps towards realisation. Services 1 and 2 need decisions on PRACE management level, Service 3 will continue the prototypal phase while Service 4 will transform into regular service.
b. Services 5–6 started from the beginning. They are aligned with the huge importance of adapting container technology for HPC and for convergence between HPC and big data. We have analysed the state-of-the-art in the area of these services, prepared operational plans for them and made first steps towards realisation. These two services are also aligned with efforts in WP4 and will provide containers with big data analysis tools for the MOOC on HPC and Big data that has been developed within WP4.

Each service is described in one section and the last section contains conclusion and orientation on the future work.

2 Service 1: Urgent Computing

2.1 Introduction and motivation

The work on the Urgent Computing (UC) service was conducted in PRACE-4IP due to the prospective and possible usage of PRACE Infrastructure in the future. It is hoped to include this service into the PRACE core service list in case of the agreement on PRACE management level. The detailed description of the service was included in the PRACE-4IP D6.4 [12] policy design document – Whitepaper [13].

We have investigated several possible scenarios and made the trade-off proposal based on the PRACE-RI usage in an Urgent 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 must state that the data staging is assured by the UC user itself directly on the selected PRACE machine (without Instrument / Storage engagement at urgent times).

The operational platform supporting UC cases 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 malfunction, or even the above noted updating of other software components, there should also be a 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 required in 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, for example, have a large enough dedicated partition for jobs with a fairly short wallclock 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 a legal memorandum between PRACE and external Urgent Computing User (Institution) will be created. Due to the merit of this work which is non-technical at this stage PRACE-4IP decided to wait for the EC review outcome in 2017. Indeed, it was discussed during the Work Package technical meeting and the official statement was presented at the PRACE-5IP All-hands meeting in 2017. However, the delivery of PRACE-4IP EC Review document [14] was delayed. The PRACE-4IP reviewers suggested that political support should be sought and this has been planned in PRACE-5IP. The future steps include the seeking of political support from PRACE managerial bodies which addresses the reviewers comment:
"Particularly the urgent computing is recognised 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" [14].

2.2 Implementation plan

The work on this service will progress along the following tasks:

Description of task	Start	Planned End	Participants
Discussion with Project Management Board and Board of Directors (AISBL) on levels of next steps	Ongoing	12/2017	PSNC, ULFME, PMO
Operational plan for PRACE UC service ready	12/2017	3/2018	PSNC, PMO, AISBL
Approval of operational plan for PRACE UC service ready	3/2018	7/2018	PSNC, PRACE AISBL
Realisation of Operational plan	7/2018	4/2019	CNRS / IN2P3, NCSA

Table 1: Planned implementation actions for Urgent computing

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

3.1 Introduction and motivation

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.

In the PRACE-4IP project, Task 6.2 partners worked on the design of a new service that will facilitate the processing and analysis of LSSI data on PRACE infrastructure. Two major needs of scientists and operators of LSSI were identified:

1 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.
2 Experimental data usually needs to be transferred from the instrument site to the external HPC facility, which requires efficient, secure, and reliable software tools and high throughput network.

In PRACE-4IP, the efforts were concentrated on the second aspect of the problems. However, one of the conclusions of the collaborations with LSSI hosting institutions was the lack of some sort of institutional access to the PRACE infrastructure. Three out of five collaborating instrument partners indicated their interest in obtaining regular access to additional compute power at the PRACE facilities. 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 have a history of collaboration with a different scientific community with different goals and using a different LSSI: CNRS/CC-IN2P3, CASTORC, NCSA. This provides a unique opportunity to draw a more generic partnership framework.

3.2 Operational Plan

Description	Start	Planned End	Participants
Identification of contacts at LSSI	Ongoing	09/2017	CNRS / IN2P3, NCSA
Project plan development	Ongoing	10/2017	CNRS / IN2P3
Specification of the needs for HPC resources by LSSI	Ongoing	10/2017	CNRS / IN2P3, NCSA
Identification of Pilot use cases	09/2017	12/2017	CNRS / IN2P3,
Implementation of Pilot use cases	01/2017	04/2018	CNRS / IN2P3, NCSA
Analysis of results	04/2017	07/2018	CNRS / IN2P3, NCSA
Definition of formal collaboration	07/2018	09/2018	CNRS / IN2P3
Report compilation	09/2018	12/2018	CNRS / IN2P3, NCSA

Table 2: Planned implementation actions for Links with Large Scale Scientific Instruments

3.3 Cooperation with ESRF

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. The real-time constraints of the computations made during data acquisition make them unsuitable for a pilot use case, as an on-site execution should be favoured. The post processing of produced data raises some challenges related to data transfer from the data production site to a HPC facility.

The collaboration between PRACE and ESRF will progress in three steps that roughly correspond to the different types of application. First, a pilot based on the pre-acquisition simulations will be implemented. It will allow the partners to demonstrate the feasibility of the collaboration between PRACE and the ESRF and to identify potential issues. Then, more challenging applications in

terms of data transfers, typically related to data reconstruction may be considered. Finally, the partners will seek among the domain specific analysis for computations that could benefit from the processing capacities of Tier-0 centres. The following table details the implementation of the first pilot.

Task	Start	End	Person in charge
Send a form on application requirements	11/09/17	15/09/17	F. Suter
Get BoD agreement on use of Tier-1 resources	11/09/17	25/09/17	F. Berberich
Send a list of candidate applications	15/09/17	29/09/17	ESRF
Select applications and contact person	01/10/17	15/11/17	ESRF
Select PRACE site offering resources	01/10/17	15/11/17	J. Povh
Deploy code on PRACE site	15/11/17	31/12/17	TBD
Testing phase	01/01/18	30/04/18	TBD

Table 3: Implementation plan for the pilot with ESFR

3.4 Cooperation with European Organisation for Nuclear Research (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.

A fundamental software in the scientific fields covered by the LHC is Geant4 (GEometry ANd Tracking) – that simulates the passage of particles through matter (https://geant4.web.cern.ch/geant4/). Geant4 simulations are very demanding in terms of computing power and generated data volumes and can benefit from HPC architectures with large clusters of Intel Xeon Phi coprocessors. Moreover, 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 order to keep the momentum and build a pilot from existing initiatives it has been decided during a joint PRACE-CERN meeting held on 21 September 2017 to organise a 1-day workshop by early

November 2017. This meeting will gather representatives from PRACE Tier-0 centres and LHC experiments. The objective is to define the scope of an on-demand training to prepare developers to target HPC resources and to determine what would be the best strategy to address the data management and access issues. The outcomes of this meeting will shape the pilot and set the milestones of its implementation.

4 Service 3: Smart post processing tools including in situ visualisation

4.1 Introduction and motivation

In this section, we will describe the pilots that we will implement for the visualisation service: the smart post-processing service, the in-situ visualisation service and the remote visualisation service. The need for the development of these tools originates from the fact that more and more time is spent during the simulation for input/output (I/O) operations and post-processing. I/O is recognised to be the main bottleneck to achieve the Exascale computing, which is up to 1000x faster than current Petascale systems.

The goal of the first pilot, called SAIO (this stands for Semi-Automatically I/O-Tuning Framework), is to reduce the time spent during the parallel I/O operations of a simulation. It is a framework that utilises a machine learning approach to optimise the parallel MPI/IO operations. The previous two pilots shared the common goal of reducing the turnaround time from simulation to insight, relying on open source technologies and trying to be as architecture-neutral as possible. They dealt with visual interactive post-processing applications and aimed to simplify their deployment and adoption and provide an environment where the computational scientist would find it easy to use visual tools to explore and analyse the output data and to steer the computations.

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

4.2 The smart post processing tool

The smart post processing tool (Semi-Automatically I/O-Tuning Framework: SAIO), which was developed by HLRS with support from PRACE-4IP, aims to optimise the parallel I/O operations semi-automatically. It was successfully tested on Hazel Hen (Cray XC40 with Lustre parallel file system) at HLRS with diverse MPI-IO, parallel HDF5 and parallel NetCDF based applications. With the help of SAIO, two data processing processes of two different user applications were accelerated by about 180% and 23%. The mechanism of the prototype's machine learning concept was proven to be efficient and scalable. In addition, SAIO was also tested on the Laki Cluster (NEC LX Series with Lustre parallel file system) at …with Intel MPI implementation and IOR benchmark at HLRS.

Since the training process of SAIO for applications and data processing processes consumes a lot of computing resources, it is worth inspecting, if the parallel I/O configuration knowledgebase found on the HPC platform A could be applied to the HPC platform B. This approach could save a lot of computing resources, which are used to searching for optimal I/O configurations, and increase the efficiency of HPC platforms.

4.2.1 SAIO

SAIO is a light weighted and intelligent framework that utilises the machine learning concept to optimise parallel MPI-IO operations. By following the MPI standard and building upon MPI-IO library, it is compatible and scalable with MPI based applications as well as portable across multiple HPC platforms. In addition, it frees the users from struggling with different I/O strategies or I/O configurations for their applications through setting the MPI info objects transparently.

Using the built-in prototypical MPI-IO training process, SAIO can find out the optimal combination of MPI info objects for specified applications by consuming some extra computing hours. These found MPI info objects build a knowledgebase for the real production process. While SAIO applies the MPI info objects from this knowledgebase, users benefit from a relatively higher I/O performance even without knowing the existence of SAIO.

SAIO works well on Hazel Hen and Laki Cluster at HLRS with a Lustre file system. Evaluating SAIO on other HPC platforms as well as other parallel file systems such as General Parallel File System (GPFS) is essential to release SAIO as an open source tool.

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 optimised by the users.

4.2.2 Operational plan for further development of SAIO

- Prototypical deployment of the tool on two additional PRACE sites
- Analysing the parallel I/O specifications of testing partner's HPC platform with the help of local support
- Evaluating SAIO with the help of local support
- Analysing the relationship of the parallel I/O configuration knowledgebase among the different HPC platforms
- Trying to find mapping mechanisms for these different parallel I/O configuration knowledgebases

4.3 In-situ visualisation pilot services

4.3.1 Description of the service

The purpose of this sub-service is to explore the feasibility of applying in-situ visualisation techniques to a widely adopted Open Source CFD simulation code. Preliminary evaluation suggests OpenFoam as a good candidate for this experiment as:

- OpenFoam has a very strong community of users, power users and developers; just in CINECA they roughly count in several tens.
- The idea of development of an OpenFoam Catalyst adaptor has been submitted to Google Summer of Code 2016. As far as we know, no in-situ instrumentation has been done yet in OpenFoam.
- ParaView is currently suggested as the main post-processing tool and already supports an efficient OpenFoam file reader, so a "poor-man" interactive traditional post-processing involving I/O could be already deployable.

- Our preliminary contacts with some of the OpenFoam developers suggest that there could be interest in ParaView Catalyst instrumentation and some work on the underlying in-memory mapping from OpenFoam data structures into VTK (Visualization Toolkit) has already started.
- CINECA is collecting example data of various sizes for testing and for future support for scaling.

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 analyse the results on the fly;
- Allow for scaling of heavy post-processing operations that would benefit from the same scaling available for the simulation.
- 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).

The environment in which this pilot could be applicable range from Tier-1 to Tier-0 machines where medium to heavy computational CFD simulations are carried out. Scripted and formalised recipes would ease the adoption.

Final users would also benefit from already available test cases, including visual post-processing workflows to flatten their learning curve of the software itself as well as lowering the barrier of using remote computing facilities.

4.3.2 Implementation plan for the service

The work on applying in-situ visualisation techniques to the OpenFoam CFD tool will consist of the following tasks:

1. Deployment of currently available components of the OpenFoam software as well as its ParaView post-processing plugin. The deployment will happen initially on CINECA clusters based on currently supported installation recipes, based on Spack package management and collected in a publicly available repository.
2. Some available test cases of increasing size will be collected and a suitable data visualisation scripts will be implemented in order to define a sound base for evaluation of possible simulation and post-processing workflows for OpenFoam community. Test execution will be automated as much as possible, scripted and collected in a versioned repository, so to provide a body of reproducible tests to compare post-processing workflow performance across different architectures and configurations (GPU, non GPU).
3. A bare-bone minimal instrumentation of OpenFoam with ParaView Catalyst will be developed, with the supervision of at least one OpenFoam developer. The implementation will be tested with at least one of the available testing workflows collected in step 2. The OpenFoam developer community will be asked for feedback as well as suggestion for proper implementation and code development policies.
4. In case of positive feedback from developers, the prototype will be extended to implement at least one of the most used post-processing techniques and the implementation will be

evaluated for performance on the test workflows in step 2. Results will be available as a prototype branch of the current OpenFoam code, to be evaluated by the community for adoption.

5. Other interested PRACE centres will be involved in deployment of the produced recipes on their infrastructures.

The following table presents a summary of actions with deadlines.

Description	Start	Planned End	Participants
Deployment of currently available components of the OpenFoam software as well as its ParaView post-processing plugin	Ongoing	12/2017	CINECA
Collection of data visualisation scripts for evaluation purposes	1/2018	4/2018	CINECA
Bare-bone minimal instrumentation of OpenFoam with ParaView Catalyst	1/2018	8/2018	CINECA
Extension of the prototype with the most used post-processing techniques	9/2018	12/2018	CINECA
Detection of PRACE centres where to deploy the produced recipes on their infrastructures	1/2019	2/2019	CINECA, other Prace centers (HRLS, CEA, …)

Table 4: Implementation plan for the In-situ visualization

4.4 Remote Visualization

4.4.1 Description of the service

A tiled visualisation tool, TileViz, 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. Instead of a single huge screen, the TileViz software allows users to display many visualisation tools outputs side by side:

http://www.maisondelasimulation.fr/Phocea/Page/index.php?id=173

Thanks to the use of Docker containers, the TileViz tool (for Tile Visualisation) is versatile. This small multiple approach (see Figure 1), coupled with remote containers on HPC computers (where the data are computed and stored), provides a very efficient tool to analyse ensemble of simulations.

Figure 1: TileViz with BrainVisa tool

Within each container we launch a X graphical server, with graphical acceleration plus a Virtual Network Computing (VNC) server (a remote access tool) and the graphical applications needed by researchers.

The containers are remotely deployed on the supercomputer processing the numerical simulations. The running applications (like Visual Molecular Dynamics - VMD) are exported to our application (running locally on the wall) through VNC. The application is also used to modify the scripts or parameters of the visualisation tool. Thanks to the VNC shared mode and QR-codes, zoomed views of any element can be displayed instantaneously.

We have used noVNC websocket package to send VNC output coming from the supercomputer (through ssh (Secure Shell) tunnelling) on the web server to display the tiled representation of the simulations.

With all the web browsers features of TileViz, we can compare two tiles by transparency or zoom. It is also possible to analyse metadata of the tiles, tag them and sort them with the tags. We can draw on the container output and save the drawings and the views.

Scientists are able to analyse multiple simulations at the same time, with varied parameters, or to visually compare similar results.

We have already a first demonstrator in Javascript for the browser GUI (Graphical User Interface) and a set of scripts which launch the containers Docker with Xvnc server with graphical acceleration on our graphical cluster Mandelbrot (with nine K5000 GPUs in five rendering nodes). We are able to duplicate 3D rendering tools, one per element of the ensembles and project the pixel fluxes on our video wall.

With all the web browsers facilities of TileViz, we can compare two tiles by transparency, print or analyse metadata of the tiles, tag them and sort them by tag, or draw on the container output and save the drawings and the views.

We want to build a complete client-server tool in conjunction with a high-level container framework (like Docker Swarm or Kubernetes) on HPC machines and their schedulers to launch

containers on these supercomputers with all security management and get the results on the high resolution video wall Mandelbrot.

4.4.2 Implementation plan for the service

We have to choose the technology of our second version of TileViz. We will use one python framework for the client-server, which can give use authentication, ssh keys management and the pixel fluxes over ssh tunnels.

We also want to test other fluxes and X11 technologies inside containers like x2go, Xpra and their insertion in HTML5 (Hypertext Markup Language) canvas inside TileViz tool.

To be able to achieve remote connections on supercomputers, we have to deal with security problems. VNC is not an easy-to-use protocol on those centres because security engineers don't really know who is connected in the custom centre. Also, we want to create hundreds of ssh tunnels with VNC connection in our tool. Containers have other security problems and virtualisation is simply not activated on almost all machine nodes.

The development and the deployment of our tool will consist of the following tasks

1 Improve our process on our local Mandelbrot cluster;
2 Deploy on Poincaré machine hosted at IDRIS computing centre;
3 Deploy in IDRIS and TGCC computing machines;
4 Deploy on any PRACE computing centres.

A work session can be planned on a desktop computer, with a browser, and then brought in the high-resolution room MANDELBROT (with 8K pixels or more in other DIGISCOPE1 network platforms as an illustration) for collaborative meetings to manipulate the tiles on the wall with our 64 inches tactile table, or any connected tablet.

We use our graphical cluster with four rendering nodes with two K5000 each and a big memory (2 TB) with one K5000 GPU to deploy Docker containers and have hundreds of tiles (it depends on the application and the dataset) on the wall.

[1] http://www.digiscope.fr/en

Figure 2: TileViz with climate simulation results on Wilder LRI video wall

We want to propose to users of local French computing centres IDRIS at TGCC to analyse their work with TileViz and Mandelbrot platform using smart connections to Maison de la Simulation.

This generic application TileViz gives the possibility to have plenty of cases to work for collaborative analysis in front of high resolution display wall. The web server in TileViz can be used with a web service and help to view some results from its database of simulation.

Some collaborative meetings have been done in the Mandelbrot room to analyse their simulations on our graphical cluster.

Furthermore, we have developed a multi-server for the Android remote control of VMD to be able to move synchronously any selected subset of the tiles. TileViz running on our 8K display wall enables you to visualise up to hundreds of trajectories during collaborative meetings.

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

5.1 Introduction and motivation

The main objective of Service 4 is the provision of repositories for European open source scientific libraries and applications, to promote wide adoption, uniformity at consolidation of European products. This task is coordinated by BSC and the other partners involved in the task are GRNET, CESGA and NIIF.

This service must provide 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 proposed solution is 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 will be deployed and

will serve as key elements to achieve the required wide adoption and uniformity. Some of these complements will consist, for example, in a project management tool, a bug tracker, a continuous integration system or a knowledge database facility.

It has been decided that the core of the service will be based on GitLab given the analysis of current technologies and possible features studied in the previous PRACE-4IP project. Other components are based also on open source software and are TRAC, Redmine, Jenkins, CASino and EUDAT B2SHARE. A first prototype was deployed in PRACE-4IP and the objective in the next months will be to move this prototype to a production service. To achieve this objective we are presenting in this deliverable a roadmap with the different tasks and milestones.

During PRACE-4IP Project, the service has been designed technically and implemented a pilot that could serve as a proof of concept for the future production service. The initial analysis of state of the art and possible tools were examined and commented in Deliverable D6.3 [1]; meanwhile technical justification of the different tools used and their recommended configuration for the repository was reported in Deliverable D6.4 [2].

As a result of all the previous work, here we are going to describe the final technical description of the service, this information has been prepared with the goal of being part of the future service user guide, one of the requirements needed before moving the service as a production system.

5.2 Final design of the PRACE Open Source repositories - User Guide

The online open source PRACE repository can be accessed from the next URL:

https://repository.prace-ri.eu/

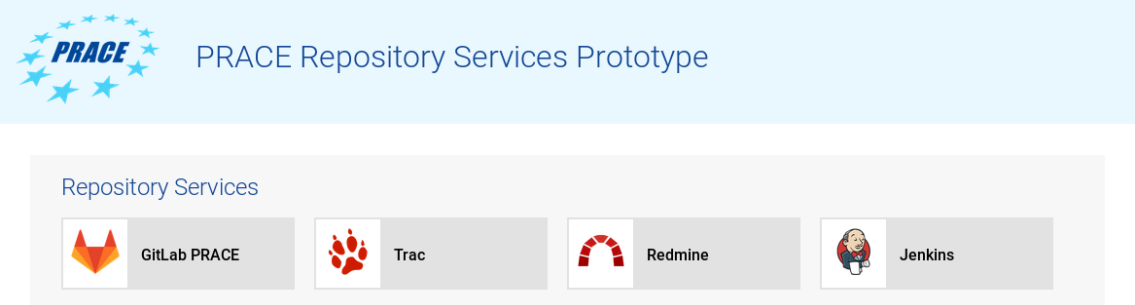

Figure 3: Repository Services website

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

You have to click on the link of any of the tool logos in the main page to be redirected to the main page of each of the tools, first of all a login page will be presented in which your repository credentials must be introduced.

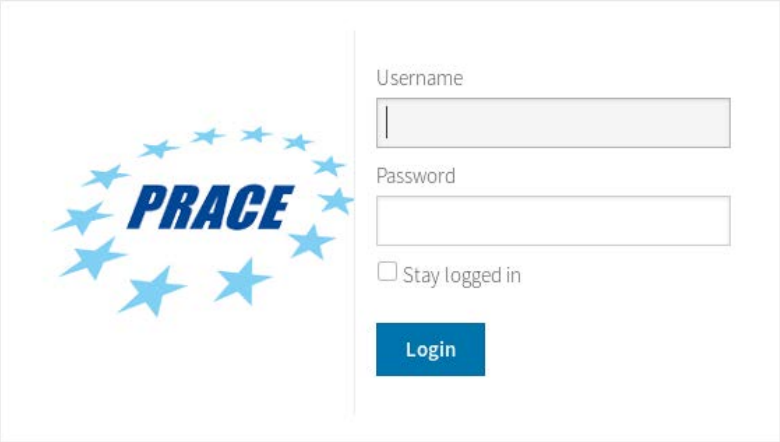

Figure 4: Login page

Login page uses 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.

Next, we are going to describe each of the tools that the PRACE repository is comprised of and provide links to some of the tutorials and material that explain their purpose and how to use them.

5.2.1 Gitlab

GitLab is a web-based Git repository manager with wiki and issue tracking features, the main page for Gitlab. In the next URL, you can find most of the information about Gitlab on [23].

We recommend you take a look at the "Getting started with GitLab" chapter which introduces most of the common concepts and usage of this tool.

Related to Gitlab, we also recommend you review Git documentation in which Gitlab is based on, you can find git information in the URL [24].

5.2.2 Trac

Trac is an enhanced wiki and issue tracking system for software development projects. Trac uses a minimalistic approach to web-based software project management. We recommend that you take a look at the User Guides which you will find in the Trac website [25].

5.2.3 Redmine

Redmine is a flexible project management web application. Its user guide can be found in the URL [26].

5.2.4 Jenkins

Jenkins 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, which compromises tasks inside Continuous integration (CI) as Continuous delivery (CD).

There is available Guided Tour for Jenkins which introduces most of the Jenkins concepts [27]. The reader can access the user guide for more detailed information also in [28].

5.2.5 B2SHARE

BoD decided the EUDAT B2SHARE service should be used to upload and store all PRACE and PRACE-related publications, creating a new 'PRACE' community. A link to the relevant advanced search from the relevant PRACE webpage is required. Initial contact with the EUDAT representative within PRACE and service provider CSC has been made. The B2SHARE guide can be found in [29].

5.3 Summary of the relevant environment and potential users

The service has been presented at the PRACE All-Hands Meeting and mailing lists, and several CoEs and projects have been contacted. Here we present an updated list of the CoEs and other partners that have been directly contacted:

- BioExcel CoE – KTH Sweden
- EoCoE Energy oriented CoE – CEA France

  EoCoE demonstrated a big interest in a GitLab solution. They are planning to deploy an account to GitLab to host private codes and extract public parts in order to make them widely available. Their interest is focused in two main aspects. First, they want a quick and easy account creation and second, they are interested in the possibility to host private projects. If both requirements can be fulfilled, they would be interested in using this service in the long term
- NOMAD Novel Materials Development – MAX-PLANCK Institute, Germany
- MAX materials design at the exascale – CONSIGLIO NAZIONALE DELLE RICERCHE, Italy
- EsiWace – Earth System Modeling for Weather and Climate – DEUTSCHES KLIMARECHENZENTRUM GMBH, Germany
- An e-infrastructure for software, training and consultancy in simulation and modelling – EPFL Lausanne, Switzerland
- POP, Performance Optimization and Productivity CoE – BSC, Spain

  POP is not interested in a code repository since the nature of its work does not require such a service, but maybe it could be useful for storing some program traces.
- CoEGSS Global Systems Science – HLRS, Germany
- PRACE Codevault

  The PRACE Codevault team has already started to use the service. Their main concern is how the access to the repository will be granted, for example to users that are not registered in the PRACE LDAP. They see the service as a great tool but they want to be sure that all the students can access the codes that are uploaded. Also, there is the concern of how to manage the accounts of project managers that has to be able to define read permissions for ones, and write permission for others on the given repositories.
- Project Alya – BSC

5.4 Operational plan

After the design of the pilot service during PRACE-4P, the remaining tasks to perform during PRACE-5IP for Service 4, consists in evolving the pilot to a PRACE production service. To achieve this objective, an operational plan has been developed to identify which tasks need to be taken into consideration or further developed from the previous work.

Here we present the subtasks that have been identified, their description and a report on the status of each one.

Task ID	Task name	Starting Date	Due date	Responsible
1	**Policies definition**			
1.1	Prepare document summarizing policies	10/02/2017	10/06/2017	BSC
2	**Server deployment**			
2.1	Pilot servers clean-up	10/02/2017	10/16/2017	GRNET
2.2	Set up consistent backup solution	10/02/2017	10/16/2017	GRNET
3	**Services deployment**			
3.1	Move Gitlab to production and announce it	10/23/2017	10/27/2017	NIIF
3.2	Move Redmine to production and announce it	10/23/2017	10/27/2017	CESGA
3.3	Move Trac to production and announce it	10/23/2017	11/30/2017	CESGA
3.4	Move Gitlab CI to production and announce it	10/23/2017	11/30/2017	CESGA
3.5	Move Jenkins to production and announce it	10/23/2017	12/31/2017	CESGA
3.6	Deploy B2Share service, adapt it to PRACE look and feel	10/23/2017	10/27/2017	NIIF
3.7	Create basic usage guide for each service	10/23/2017	12/31/2017	ALL

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

5.4.1 Subtask 1 – Policies definition

An initial policies definition for a future repository service was discussed during PRACE-4IP, this subtask will take care of and collect those thoughts to prepare a formal document summarising the policies of the future production service.

The policies document prepared during this task will be distributed to the PRACE operations team, to be validated and make sure it is coherent and consistent with other PRACE production services and whole PRACE-RI as a whole.

Policies definition document will include:

- Who can request access to the repository
- Procedure to request access to or resources in the repository
- Initial Usage use policy and terms and conditions of the service

5.4.2 Subtask 2 – Production server deployment

During all the service prototyping and analysis of the possible tools, the same virtual machine has been used. After analysis, this virtual machine cannot be used as the production server, as it holds a lot of temporary software and configurations that has now been decided not to use.

An exception is the B2SHARE service, which is a completely external service hosted by EUDAT.

One of the main aims of this subtask will be to deploy a new and clean server for the production repository service.

This subtask will also take care to review the initial resources (vCPU, memory and storage) that the production server needs to have. The configuration of the production server has to permit the reconfiguration of these resources, so that the service can be flexible enough to adapt to future user requirements.

The last aim of this subtask will be to design a backup and/or High Availability solution for the production server deployment. First, a backup of any of the database and data has to be provided via a backup system to another storage device, and a disaster recovery procedure has to be in place.

A study needs to be done in order to determine which level of availability is needed for the future repository service, the team has envisaged 3 types of high-availability solutions, ordered from minor to major protection:

1 No HA (High Availability) solution
   - Advantages: immediate
   - Drawback: Risk of outages of the service in case of failure of the server

2 HA solution implemented on one site
   - Advantages: A little investigation needs to be done how to implement the high-availability: active-passive or active-active methodology.
   - Drawback: Risk of outages in case of complete site failure

3 HA solution implemented on multiple sites
   - Advantages: Robust
   - Drawback: Risk in the synchronisation of data between different sites, and further investigations on how to achieve it. The service grows in complexity.

Further discussion will follow inside the team and future users inside PRACE infrastructure, namely WP4 and WP7, to decide which level of availability this service really needs.

5.4.3 Subtask 3 – Services deployment/configuration/testing

Once subtask 2 finishes, the server is deployed and backup and a HA policy is in place, a reinstallation of the different tools that forms the services will be done on the new server, configured and prepared to provide production-quality service.

Inside this subtask a basic user guide will be prepared so future users of the service can understand the different components and be directed to public tutorials of those components.

Finally, any procedure that needs to be done in order to define each of the components of this service as a PRACE production service will be performed inside this sub-task. Tight coordination and collaboration with Task 6.1 inside the PRACE operations Work Package (WP6) is expected.

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

6.1 Introduction and motivation

Sharing of software tools is an essential demand among scientists and researchers in order to reproduce results. HPC centres are struggling to keep up with the rapid expansion of software tools and libraries. In some cases, large communities are developing software to serve their specific scientific community. In many cases, users may be interested in tools that are difficult to install, due to a long list of non-portable dependencies. Some requested software might be specifically targeted at an OS environment that is common for their domain but may conflict with the requirements from another community. For example, the biology and genomics community may adopt Ubuntu as their base OS with a specific version of Perl and Python [3].

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 [2] 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. Singularity [4] is another container engine that is mainly targeting Linux containers on HPC platforms. It is installed on many HPC clusters in production.

With the use of Linux containers, researchers can install tools on their platform of choice, e.g. Ubuntu for bioinformaticians and CentOS for physicists, as e.g. a Docker image and publish it on the Docker hub or just share the Dockerfile with collaborators. Then anyone who has a Docker engine and a proper Linux kernel may run the image and get the same functionality. This makes life easier for software developers in that they no longer need to write multiple installation guides and test on different Linux distributions. It also makes life easier for system administrators, as instead of receiving software requests of type: "I need software X, and here is the installation guide, please install it!", requests will be of type: "I need software X, here is the name of its Docker image, please pull it". In addition, no software maintenance will be needed and no dependency conflicts will arise when installing new software [1]. HPC targeted platforms, e.g. Singularity and Shifter, 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 [20] Each VM contains its own operating system, known as the guest OS, on the top of which software packages are installed. 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 Site contributions

The following is a list of the PRACE sites contributing to service 5:

- UiO/SIGMA2 (Coordinator)
- CEA
- CINECA
- CNRS/IDRIS
- EPCC, CESGA: observers

The following is a list of HPC clusters and platforms assigned by each site for service 5:

- UiO/SIGMA2:
  - 2 Slurm clusters (Abel and Colossus) with CentOS 6.8 compute nodes and BeeGFS file-system. Colossus is the HPC cluster for TSD (Services for Sensitive Data).
  - Slurm cluster (Fram) with CentOS 7 compute nodes and Lustre file-system.
  - HTCondor (High Throughput Condor) test cluster on the Norwegian UH-IaaS OpenStack cloud with 30 CentOS 7 compute nodes and NFS file-system.
  - 100,000 CPU hours on the Cray cluster at CSC
  - Moab cluster with 540 CentOS 7.3 compute nodes and EMC ISILON storage at Computerome
- CEA:
  - Slurm cluster with Red Hat Enterprise Linux (RHEL) 7.3 compute nodes and Lustre file-system
- CINECA:
  - Portable Batch System (PBS) cluster with CentOS 7.2 compute nodes and GPFS file-system (two nodes can be used for the task).
  - OpenStack cloud
- CNRS/IDRIS:
  - IBM Spectrum Load Sharing Facility (LSF) cluster with RHEL7.3 and GPFS file system (prototype machine)

- CESGA:
  - Slurm cluster with CentOS 6.8 compute nodes and NFS file-system. Lustre file-system is used in addition to store the container images.
- EPCC:
  - PBS cluster with 280 CentOS 7 compute nodes and DDN Lustre file-system.

6.3 Operational plan

The operational plan for service 5 is briefly described as follows:

- Collect interests: Which sites are interested and which queuing systems.
- Compose a list container and VM platforms to be evaluated.
- Compose a list of potential use cases and user communities/groups who will benefit of these use cases.
- Install the targeted container and VM platform on the available suitable HPC clusters provided for service 5 for testing, and start testing and evaluating the use cases.
- Generate reports on:
  - Usability (How the amount of work is reduced for sys-admins to deploy new software, and how users can easily access and run new applications on HPC clusters).
  - Security (How secure are container and VM based packaging compared to traditional native installation)
  - Performance (investigate whether there are performance degradations, and whether the levels are acceptable)
- Compose a secure and stable workflow for users to run their own container and VM images on the system, and to convert e.g. Docker images to other formats (e.g. Singularity and Shifter).
- Create a PRACE repository on Docker-hub (or private registry) for PRACE maintained data analytics and visualization images.
- Compose a PRACE policy for sys-admins on how to create in-house container and VM images (maintained by the system, e.g. MPI images), and how to configure shared image repositories.
- Define and test a collection of front-end solution for submitting, monitoring, and managing container/VM jobs on HPC cluster (e.g. portals and meta-schedulers).

The following table presents a summary of actions with deadlines.

Description	Start	Planned End	Participants	Current status mid Sep 17
Get the project plan ready	Ongoing	10/2017	SIGMA2, CEA, CINECA, IDRIS, CESGA	Prepared and is under review
Define a list of use cases [2] that require containers or virtualised environment on HPC	Ongoing	10/2017	SIGMA2, CEA, CINECA, IDRIS, CESGA	10 use cases collected locally on sites, and 15 use cases collected through a web form
Testing and evaluation of virtualised environment on HPC (PCOCC, HTCondor, others)	Ongoing	12/2017	CEA, SIGMA2	Docker, Singularity, PCOCC, HTCondor have been tested for serial and parallel jobs
Testing and evaluation of container and VM orchestration tools within HPC (PCOCC, Galaxy, UNICORE, others)	Ongoing	05/2018	CEA, SIGMA2	PCOCC and Galaxy are installed in CEA and UiO and the testing is ongoing
Test and evaluate existing container platforms on the HPC infrastructures (in terms of security and performance). [3]And develop new tools when needed	Ongoing	03/2018	SIGMA2, CEA, CINECA, IDRIS, CESGA	Docker, Singularity, and PCOCC are under testing. PCOCC [40] and Socker [1] are in house tools developed by CEA and UiO
Integrating the selected platform(s) into the test[4] environment on each site and test the selected tools	03/2018	09/2018	SIGMA2, CEA, CINECA, IDRIS, CESGA	Planned
Reporting of pilot and deliverables	09/2018	12/2018	SIGMA2, CEA, CINECA, IDRIS, CESGA	Planned

Table 6: Planned implementation actions for HPC containers pilot

6.4 Prototypes and use cases

Prototypes are classified according to the target into:

- Virtual machines, VMs
- Linux containers
- Portals for submitting container jobs to HPC cluster

In the following subsections, each prototype is described.

6.4.1 pcocc: Private Cloud on a Compute Cluster (VM prototype)

pcocc (pronounced like "peacock") [40] allows users of a HPC cluster to host their own clusters of VMs on compute nodes alongside regular jobs. Users are thus able to fully customise 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

[2] This action will be in collaboration with service 3 (visualization) and with service 6 (big data analytics). Likewise, with WP4 with creating appropriate containers for MOOCs

[3] The target is to decide on what can be implemented in large scale HPC systems.

[4] It has to be taken into account that some systems might not be so flexible as a test environment in terms of deploying new platforms, etc.

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. pcocc has been developed at CEA and is currently deployed in production.

6.4.1.1 Preliminary tests

Figure 5: depicts the relative execution time of common benchmarks executed in a cluster of pcocc VMs compared to the same benchmarks launched on the host cluster. The nodes are composed of dual Broadwell CPUs (28 cores total) clocked at 2.4 Ghz and 128 GB RAM. We can see that the virtualisation overhead remains between 0 and 5%.

6.4.1.2 Potential use cases

At CEA, different use-cases are implemented in production:

- **Genomics user**: This population need a lot of brand new specific products with always the last version. For a computing centre, it is very difficult to provide such an up-to-date environment. CEA provide a solution based on virtualisation: it is not efficient but easy to implement and provide a usable solution.
- **Industrial user**: Some industrial users have certified codes which are validated on a specific OS (like Windows for example). Virtualisation can let us run such codes, or pre/post processing tools.
- **Admin user**: Testing in advance new OS version before putting it in real production is very convenient for administrators.
- **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.

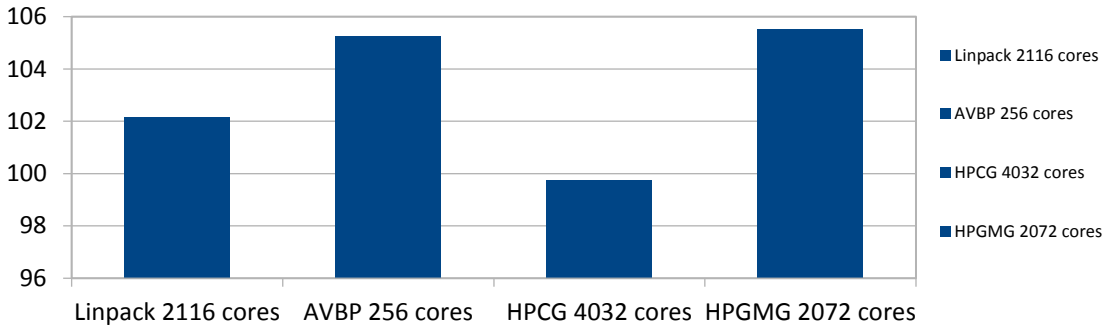

Figure 5: Relative execution time of common benchmarks executed in a cluster of pcocc VMs compared to the same benchmarks launched on the host cluster.

6.4.2 Socker: running Docker containers on Slurm (Container prototype)

Socker [1] is a wrapper for running Docker containers securely inside HPC jobs. It is mainly designed to enable the users of our Slurm clusters (Abel and Colossus) to run Docker enabled jobs. Since our compute nodes are running CentOS 6.7, which has the standard kernel 2.6, Socker is designed to be able to work with Docker 1.7 (since support for CentOS 6 is dropped for newer

Docker releases). It can also deal with newer Docker versions. 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. To enable Socker on HPC clusters, Docker 1.7 has been installed on compute nodes on the main cluster at UiO, Abel [15]. Currently Docker doesn't support shared image repositories among hosts, so each compute node has its local image repository (layered file system).
To evaluate the performance of Socker for HPC applications, we performed an experiment on our main Slurm cluster, Abel. The aim of this experiment was to indicate whether Socker introduces considerable computational overhead. We used one image from Docker hub, genomicpariscentre/bowtiel, for running sequence alignment using Bowtie [6]. The input dataset used is approx. 5 GiB Fastq file, to be aligned against human genome hg19. We used the pMap package [7] to divide the alignment into parallel MPI jobs. Three compute nodes (2*16 + 20 cores) were reserved for this Docker test. The experiment included the following:

- Alignment with pMap using Native Bowtie binary.
- pMap running the Bowtie Docker image using Docker run as a system administrator account that is a Docker group member.
- pMap running the Bowtie Docker image using Socker and a regular user account.

The Bowtie image has been pulled on the three nodes in advance, to avoid any additional latency pulling the image from the Docker hub. All Docker runs (both direct Docker and Socker) were configured to remove the container after exiting. Figure 6: Total run time of parallel pmap MPI jobs using bowtie aligner as: Bowtie native binary, bowtie Docker image using Docker run, and bowtie Docker image using Socker presents the total run time against the number of parallel processes. 1, 8, 16, and 32 processes were used. It is clear that running container jobs (for both Docker and Socker) introduces very small overhead compared to running native jobs. It is noticed that the overhead grows with increasing the number of parallel processes. This is resulted from: 1) the overhead of starting new containers, 2) Removing exited containers is enabled, which introduces additional overhead (but is necessary for cleaning up and saving disk space on compute nodes). It is also noticed that Socker overall introduces almost no additional overhead compared to Docker run.

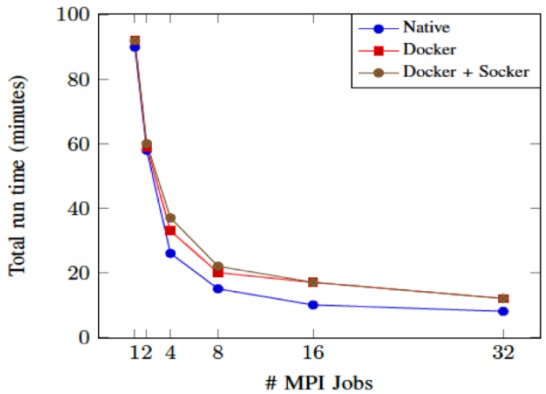

Figure 6: Total run time of parallel pmap MPI jobs using bowtie aligner as: Bowtie native binary, bowtie Docker image using Docker run, and bowtie Docker image using Socker

6.4.3 LSF/Docker platform at IDRIS (Container use case)

Docker is an interesting technology as it offers an easy way for users to run applications on various operating systems. However, the deployment of Docker at IDRIS is 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 allows users to run Docker containers in a restricted environment but offers better security. 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 *authorised* Docker images. The registry has been set up on a management node.

6.4.3.1 How the containers are executed

Spectrum LSF has been configured using the specific application profiles feature that defines applications along with the runtime behaviour and scheduling of LSF for jobs associated with a given application. The following lines give an example of the application "powerai":

```
# # This section defines the powerai docker wrapped application
Begin Application
CONTAINER = docker[image(registry:port/powerai) options( …  )]
NAME = powerai
End Application
```

This example describes the "powerai" application which is defined as a container, here a Docker one. The related image is pulled from the local registry. Any option used to run a container can be configured here such as the file system mappings, the devices that have to be used inside the container.
To be able to run a powerai application, first the user must load the powerai environment using a module command. Then, the user must specify in their submission script, the container request through the following and mandatory Spectrum LSF directive:

```
#BSUB -app powerai
```

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.

6.4.3.2 LSF/Docker Platform Security tests

The LSF/Docker platform security tests can be split into 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 practices that are usually applied in an HPC site.
- The second one 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.

6.4.4 Singularity (Container platform)

Released officially for the first time in 2016, Singularity [4] (now at version 2.3.1) is a container paradigm, designed for HPC platforms, that aims to reach three specific goals: reproducibility of results, mobility of compute and user freedom.

To reach these goals, Singularity developers were able to make that the container itself a single file image that can be easily copied or transferred from one machine to another. In such an image file, the operating system, the packages, the libraries, the software and all that is needed are packaged together in a single file. Moreover, these Singularity containers can interact with files local to the container in the directory where the container is executed (binding option).

Since Singularity containers are designed for HPC platforms, it is possible in a very simple way to run pure MPI or hybrid (MPI + OpenMP) applications inside a Singularity container, Also GPU support is available in these containers, by installing the needed driver and toolkit.

Applications which run in a Singularity container, run with the same "distance" to the host kernel and hardware as natively running applications. Singularity launches the container as the calling user in the appropriate process context. There is no root daemon process and no escalation of privileges within the container, as in the case of Docker container.

Even if Singularity is starting to be largely used in a very short time, the most popular container system today is arguably Docker. This is not an issue, it's not necessary to convert previously created Docker containers to Singularity containers, because Singularity is designed in such a way so that it is easy to use a previously created Docker container simply by bootstraping the Docker container inside Singularity.

6.4.4.1 Support for MPI

Singularity containers can be created for any application and are not just limited to HPC applications. However, one of the most positive things that Singularity can do is create simple containers for MPI and MPI+OpenMP applications. These have been very difficult for other container systems to handle, but Singularity treats the applications like an executable, so the "mpirun" command treats it like any other executable.

Singularity is still developing, but it can already be used to create and run containers for large MPI applications. The main issues are related to the interoperability version that it isn't supported in openmpi 2.x So actually, until the version 3.x will be released officially, the same version of openmpi in the Singularity container and in the host have to be installed. Moreover, for large MPI applications running on more than one node, the needed driver for interconnection network, as infiniband, has to be installed inside the container.

6.4.4.2 Preliminary tests in CINECA with Singularity

On the GALILEO cluster in CINECA preliminary tests with Singularity containers have been performed using Quantum Espresso [41], and Caffe [16] codes.

Quantum Espresso

Quantum Espresso [41] is an integrated suite of Open-Source computer codes for electronic-structure calculations and materials modelling at the nanoscale. It is based on density-functional theory, plane waves, and pseudopotentials.

In the test performed, on 64 water molecules has been made 200 steps of a car-parrinello quantum dynamics.

The version of Quantum Espresso used is 6.1.0, the version of openmpi is 2.1.1, and the compiler used for them and for the prerequisites libraries is gnu 4.8.2.

This test has been run on one node of GALILEO, using 16 mpi-processes. The execution time inside and outside the Singularity container, is more or less the same. So, no poor performance is noted using the Singularity container.

Quantum Espresso version 6.1.0 + Openmpi 2.1.1 + gnu compiler	Inside Singularity Container	Bare Metal
1 GALILEO node, 16 MPI processes	24m51.51s CPU 25m22.07s WALL	24m50.68s CPU 25m22.14s WALL

Table 7: Quantum experiment using Singularity and native

Caffe

Caffe [17] is a deep learning framework made with expression, speed and modularity in mind. It is developed by the Berkeley Vision and Learning Center (BVLC) and community contributors [16]. Release 1.0.0 exploits recent Pascal series GPUs thanks to CUDA 8.0 and CUDNN 6.0. Multi GPU (data parallel model) training is supported.

Two different tests have been performed:

- LeNet [9] on MNIST [15], the handwritten digit recognition dataset containing 60.000 training examples and 10.000 test examples.
- AlexNet [10] test benchmark, part of a suite [22] which tests all publicly accessible implementations of convolutional networks.

For these preliminary tests, the version of Caffe used is the 1.0.0 BVLC, with Intel MKL low-level library has been used, without involving the GPUs. Both tests have been run on one node of GALILEO, using 4 OpenMP threads.

In the LeNet test, 10.000 iterations are performed, and the accuracy test is computed each 500 of them. As written in the table, the execution time inside the container is 25.5 % slower with respect to the version outside the container. In the AlexNet test, both forward (fwd) and forward-backward (fwd/back) number of images per second are considered, between square brackets is the batch size. As written in the table, the number of images computed per second is more than 30% slower with respect to the version outside the container. The loss of performance maybe due to differences in the version of installed libraries, but it is currently under investigation.

LeNet	Inside Singularity Container	Bare Metal
1 GALILEO node, 4 openmp threads.	6m 23sec WALL	5m 5sec WALL

Table 8: LeNet experiment using Singularity and native

AlexNet	Inside Singularity Container		Bare Metal	
1 GALILEO node, 4 openmp threads	Imgs/s (fwd/back) 17.683 [bs: 4096]	Imgs/s (fwd) 66.758 [bs: 4096]	Imgs/s (fwd/back) 40.528 [bs: 4096]	Imgs/s (fwd) 99.004 [bs: 4096]

Table 9: AlexNet experiment using Singularity and native

6.4.4.3 Experience of MSO4SC project with Singularity

The experience of MSO4SC (Mathematical Modelling, Simulation, and Optimization for Societal Changes with Scientific Computing) with Singularity containers for HPC is reported in ref [34]. In the context of the MSO4SC project, Singularity bootstrap definition templates are provided for helping users and developers to create Singularity images with InfiniBand drivers and an OpenMPI version supported by FT2. These templates are available in ref [35] above.

6.4.5 Galaxy (Portal)

Galaxy [36] is a web portal 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 [37] . Galaxy is installed in production at the UiO [38] and the container support is under testing.

6.4.6 Advanced Resource Connector - ARC (Portal)

ARC [39] is a meta-scheduler that enables sharing and federation of computing and storage resources distributed across different clusters. ARC uses a common job description language for writing job submission scripts, that is converted to the job description language of the targeted queuing system of the HPC cluster to which the job is allocated. ARC recently added support for Docker and Singularity containers. It is currently under testing at the UiO.

6.5 Collected use cases

Service 5 has recently published a web-form for collecting research use cases for containers and VMs in HPC vii. This web-form has been published by the University of Oslo on July 2017, and made available to the EU scientific communities. The use cases were collected until September 2017. The following is a brief description of the 15 use cases collected so far (those are newly collected use cases, different from the production use cases described in the prototype sub-sections in section 4):

- **Name of the software:** caffe, ROS, gazebo, FEniCS, mriqc, freesurfer, heudiconv, upc,upc++, gasnet intel PCM (performance counter monitor), GAMBIT, FEniCS, ARMplusplus, anvi'o, GAMBIT, Ubuntu (OS).
- **Purpose of the software:** Data analytics: 7 (46.7%), Virtualization: 3 (20%), Deep learning: 4 (26,7%), Machine learning: 1 (6.7%), and Other: 5 (33%)
- **Type of packaging:** Virtual Machine: 5 (33.3%), Containers: 10 (66.7%).
- **Support for parallelisation:** Yes: 13 (86.7%), No: 2 (13.3%)
- **Kind of parallelisation:** Shared memory: 9 (60%), Distributed memory: 9 (60%)

- **The approximate number of researchers using the software:** Less than 5: 1 (6.7%), Between 5 and 20: 10 (66.7%), More than 20:4 (26.7%)

The following can be concluded from the results:

- The majority of use cases are in favour of containers.
- Most use cases are for software that supports parallelisation.
- About four use cases are useful for more than 20 researchers.

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

7.1 Introduction and motivation

Data analytic applications help scientists on discovering and extracting information from previously collected data, with a descriptive or predictive aim. This process is challenging both on computation and data management point of view: it's not only a task that requires relevant computational power, but the efficiency on data analysis is strongly affected by the architecture and technologies involved in data storage and retrieval.
Current trends show the rise of open source frameworks either for general-purpose computing such as Apache Spark [42], or for a dedicated artificial intelligence field as Deep Learning. In this second group, TensorFlow [43], Caffe [16], Torch [44], Theano [45] and Neon [46] are among the most adopted.
Apache Spark is a general-purpose cluster computing system used for processing large datasets. A number of solutions are available for improving the performance of Spark in HPC environments: RDMA (Remote Direct Memory Access)-based package from HiBD [47] offers native Infiniband and RoCE (RDMA over Converged Ethernet) support for RDMA-enabled interconnects. Different I/O solutions can also be added in order to adapt existing Spark clusters to HPC environment such as Hadoop Distributed File System (HDFS) [48] connector for GPFS or additional burst buffers.
Deep Learning involves a relevant amount of matrix multiplications, convolutions, n-dimensional filtering and similar operations that are well suited to be accelerated using GPU architectures: all major tools offer GPU acceleration out of the box. Recent interest is in many-core approach: during the last year branches of common tools (Theano, Caffe) have been directly developed by Intel [49], and IBM [50]. Code compatibility and efficiency improvement are the two main targets of these new releases, based on Intel multicore and many-core, and IBM OpenPOWER [51] architectures respectively. Multi-node training is possible either using gRPC [52] (TensorFlow), or by exploiting recently developed libraries such as MLSL [53], and IBM DDL [54] that relies on approaches more common on HPC clusters like MPI [55].
Latest trends on Machine Learning and Deep Learning software development aim at supporting resources that are already available on HPC clusters. These new features are brought to end users either through branches of popular Software Development Kit (SDK), or directly through plugins. In order to provide an HPC based data analytics service it is of primary importance to test how efficiently and reliably those new approaches exploit hardware resources.

Besides standard datasets, that can be downloaded and used as preliminary validation benchmark of the setup, this project aims at assessing the feasibility of the service through the identification of prototype use cases.

In synergy with Service 5, the proposed software list can be used to select candidates for containerised applications and collected benchmarks can be used as a reference for assessing virtualisation overhead.

Outcomes of this service can impact users from different disciplines that are interested in data analytics and are already familiar with software solutions described above.

7.2 Current status

An interest group composed of CINECA, CNRS/IDRIS, CNRS/IN2P3, KTH, SNC, EPCC has been set up. Each partner has identified one or more infrastructure that can be used for both the installation and test phases, a list of already installed packages can help to describe the initial status of this task.
CINECA:

- Infrastructure 1: **type:** production cluster, **architecture:** x86_64, **accelerators:** no, **environment configuration:** Environment Modules, **LRMS** (Local Resource Management System)**:** Altair PBS Professional**, compilers:** Intel, GNU, **libraries/applications:** Caffe, Theano;
- Infrastructure 2: **type:** prototype cluster, **architecture:** OpenPower 8 processors with NVLink bus, **accelerators:** GPU (Nvidia Tesla P100 SXM2), **environment configuration:** TBD, **LRMS:** TBD, **compilers:** GNU, IBM XL, PGI, **libraries/applications:** PowerAI, TBD;
- Infrastructure 3: **type:** production cluster, **architecture:** x86_64, **accelerators:** GPU (Nvidia Tesla K80), **environment configuration:** Environment Modules, Singularity containers, , **LRMS:** Altair PBS Professional**, compilers:** GNU, Intel, **libraries/applications:** Caffe, Theano, Tensorflow;

CNRS/IDRIS:

- Infrastructure 1: **type:** prototype cluster, **architecture:** OpenPower 8 processors with NVLink bus, **accelerators:** GPU (Nvidia Tesla P100 SXM2), **environment configuration:** Environment Modules, **LRMS:** Spectrum LSF to run PowerAI jobs within Docker containers, **compilers:** GNU, IBM XL, LLVM, PGI, **applications/libraries:** PowerAI framework (TensorFlow, Torch, Theano, Caffe);
- Infrastructure 2: **type:** production cluster, **architecture:** x86_64, **accelerators:** none, **environment configuration:** Environment Modules, **compilers:** GNU, Intel, **libraries/applications:** none;

CNRS/IN2P3:

- Infrastructure 1: **type:** production cluster, **architecture:** x86_64, **accelerators:** GPU (Nvidia Tesla K80), **environment configuration:** Environment Modules, LRMS: Univa Grid Engine, **compilers:** GNU, **libraries/applications:** Keras, Theano, Tensorflow;

PSNC:

- Infrastructure 1: **type:** production cluster, **architecture:** x86_64, **accelerators:** none, **environment configuration:** Environment Modules, **LRMS:** Simple Linux Utility for Resource Management (SLURM), **compilers:** GNU, Intel, **applications/libraries:** none;

EPCC:

- Infrastructure 1: **type:** production cluster, **architecture:** x86_64, **accelerators:** none, **environment configuration:** Environment Modules, **LRMS:** TBD**, compilers:** Cray, **applications/libraries:** Spark, Hadoop, R, Cray Graph Engine (not currently in production but expected to be available for the tasks identified).

A list of software candidates has been produced: for each item we gathered main features, prerequisites, possible interactions with HPC hardware. In order to track down the impact on users, we collected the list of groups/projects that already exploit that particular framework; moreover, since we are interested in adequately supported tools, we tracked both first and latest releases: this information can help on characterise the tool in terms of project maturity and support. Results of this preliminary information collection are summarised at the end of this section.

7.3 Operational plan

Here we present the operational plan of service 6 activities for the whole project duration period.

Description	Start	Planned End	Participants
Project plan development	03/2017	10/2017	CINECA, CNRS/IDRIS, CNRS/IN2P3, KTH, PSNC
Definition of a list of Deep learning tools	06/2017	10/2017	CINECA, CNRS/IDRIS, CNRS/IN2P3
Definition of a list of solutions to optimize Spark in HPC environments[5]	06/2017	10/2017	CNRS/IDRIS, KTH
Collection and selection of pilot use cases	06/2017	02/2018	CINECA, CNRS/IDRIS, CNRS/IN2P3, KTH
Training activity	06/2017	06/2018	CINECA
Installation and preliminary test: standard datasets	09/2017	05/2018	CINECA, CNRS/IDRIS, CNRS/IN2P3, KTH, EPCC
Tools evaluation: use cases	04/2018	08/2018	CINECA, CNRS/IDRIS, CNRS/IN2P3, KTH, EPCC
Report compilation	09/2018	12/2018	CINECA, CNRS/IDRIS, CNRS/IN2P3, KTH, EPCC

Table 10: Planned implementation actions for Data Analytics pilot

[5] This includes memory and I/0 aspects and doesn't aim to analyse the full list of available tools

7.4 Prototypes

7.4.1 Spark tools

Apache Spark [42] is an open source parallel processing framework for running large-scale data analytics applications across clustered computers. It provides an optimised in-memory data analysis engine that supports general computation graphs. It also supports a rich set of higher-level tools including Spark SQL for SQL and DataFrames, MLlib for machine learning, GraphX for graph processing, and Spark Streaming for stream processing. The main component, Spark Core, is a faster and more flexible alternative to MapReduce. It provides distributed task transmission, scheduling and I/O functionality.
Spark is frequently used to process data from the Hadoop Distributed File System (HDFS), from NoSQL databases or relational data stores, such as Apache Hive. Official website [56] includes exhaustive documentation, both programming and deployment guides, APIs documentation. It also includes examples where the various APIs are shown. A number of dedicated benchmarks are available: spark-bench [57], intel Hadoop/HiBench [58], A Big Data Benchmark Suite, ICT, Chinese Academy of Sciences [59].
Spark requires a workload manager to work (spark standalone, yarn, mesos) which needs to interface the batch scheduler. This can be done using some additional scripts (LSF).

7.4.1.1 IBM Spectrum Scale (GPFS) support for Hadoop

This commercial IBM data analytics solution [60] offers an alternative to the Hadoop Distributed File System (HDFS) for Hadoop data analytics services. The Spectrum scale HDFS Transparency Hadoop Connector is the second generation of IBM GPFS Hadoop Connector. It provides a HDFS compliant Application Programming Interface (API) and a shell command interface that allows applications to use HDFS Client to access a GPFS through HDFS RPC requests. It integrates both the NameNode and the DataNode services and responds to the request as if it were HDFS.

7.4.1.2 RDMA for Apache Spark (HiBD)

RDMA for Apache Spark [47] is a high-performance design of Spark over RDMA-enabled Interconnects. It consists in a preconfigured and modified deployment on Spark, integrated with a dynamically loaded native library which allows using InfiniBand and RoCE during several communication phases of Spark (e.g. shuffle). This version of RDMA for Apache Spark 0.9.4 is based on Spark 2.1.0, it offers native InfiniBand and RoCE support at the verbs level for Spark: RDMA-based data shuffle, SEDA-based shuffle architecture, efficient connection management and sharing, non-blocking and chunk-based data transfer, off-JVM-heap buffer management.
Official website includes the user guide [61], documentation includes instructions to run OHB micro-benchmark, Sort, Terasort, and PageRank.

7.4.2 Deep Learning SDK

7.4.2.1 Torch

Torch [44], developed by scientists from Facebook, Twitter, and Google DeepMind, defines itself as a scientific computing framework with wide support for machine learning algorithms. It features a powerful N-dimensional array type, routines for linear algebra and matrix operations, and

numeric optimization algorithms. Although all the most popular frameworks support GPUs, Torch explicitly targets these devices in its own definition, “putting GPUs first”.
The official website includes APIs documentation [62] and a (reduced) collection of tutorials. Github repository includes community-contributed projects and models [63].
Torch is based on Lua, a dynamically-typed multi-paradigm scripting language, and an underlying C/CUDA implementation. This framework is easily extensible thanks to a good integration with C. New generic packages are often very rapidly implemented by the community. On the other hand, Lua and LuaJIT are not as extended as Python, and new users may face the need of learning a new language. Nevertheless, while Lua is not Python, there are similarities between both languages, and there are several attempts to migrate Torch to python (see pytorch).

7.4.2.2 Intel Data Analytics Acceleration Library (DAAL)

The Intel Data Analytics Acceleration Library (Intel® DAAL [64]) is the library of Intel Architecture optimised building blocks covering all data analytics stages: data acquisition from a data source, pre-processing, transformation, data mining, modelling, validation, and decision making. To achieve best performance on a range of Intel processors, Intel DAAL uses optimised algorithms from the Intel Math Kernel Library and Intel Integrated Performance Primitives.
Intel DAAL supports the concept of the end-to-end analytics when some of data analytics stages are performed on the edge devices (close to where the data is generated and where it is finally consumed). Specifically, Intel DAAL Application Programming Interfaces (APIs) are agnostic about a particular cross-device communication technology and therefore can be used within different end-to-end analytics frameworks.
Unlike other libraries targeting Machine Learning and Data Mining domains, Intel DAAL optimises the entire workflow, from data acquisition from SQL and no-SQL data sources to data transformations to data analysis, training and prediction.
The Algorithms component of the Intel® Data Analytics Acceleration Library (Intel® DAAL) consists of classes that implement algorithms for data analysis (data mining), and data modelling (training and prediction). This library contains implementations of Neural networks layers for deep learning applications, as well as optimization algorithms for training. The Intel DAAL algorithms support the following computation modes: Batch processing, Online processing, Distributed processing. Intel DAAL is licensed under Apache License 2.0 [65]. The documentation is available at [66]. Installation package contains a number of examples (source codes and input data sets) ready to compile and run on the system.

7.4.2.3 Caffe

Convolutional Architecture for Fast Feature Embedding (Caffe [16]) is a C++ library released under BSD-2 license, developed by Berkeley AI Research (BAIR) and by community contributors. This SDK is widely adopted both in academic research and large scale industrial applications (Facebook, NVIDIA, Intel, Sony, Yahoo! Japan, Samsung, Adobe, A9, Siemens, Pinterest, Embedded Vision Alliance). Data Layers can read input from row images, databases, HDF5 files or directly from memory. Vision Layers implement tasks commonly encountered in imaging (convolution and deconvolution, pooling, spatial pyramid pooling, crop) while Recurrent Layers introduce the concept of memory (Recurrent Neural Network - RNN, Long Short Term Memory - LSTM). A number of activation layers are implemented, and common loss functions are present.

A short number of examples are available on SDK source, while BVLC web page contains tutorials and examples for both command-line and Python API (Jupyter notebooks), benchmarks are not directly provided by the project, but the SDK is considered in [22].
Most simple approach involves the definition of a neural network through the compilation of a plain text file containing a protocol buffer schema; solver parameters are defined in a secondary text file. Both files are then fed to main Caffe executable for a train procedure. Learned model is then serialised as binary protocol buffer and can be exported, along with the definition of the network, in one of the supported architectures, even mobile phones, for inference. Developers and deep learning researchers can also exploit the flexibility of C++ interface. Higher level API exploits Python language: notebook examples contain a step-by-step description for common tasks.
Due to it's open source (BSD) licence, this SDK counts a relevant number of forks; most notable ones are:

- IBM-Caffe from the PowerAI development team [67] contains performance enhancements aiming at leverage NVIDIA NVlink bandwidth on Power8 systems.
- Intel Caffe [68] is dedicated to improving BVLC original code when running on CPU (in particular Haswell, Broadwell and Xeon Phi).

It is worth noting that Caffe, from version 1.0, is in maintenance mode: development will continue with upcoming 1.1 version, but it will be placed side by side with a new development line: Caffe2 [69].

7.4.2.4 Theano

Theano [45] is a Python library for efficient evaluation of expression containing multi-dimensional arrays. Fast C code can be dynamically generated and compiled for either CPU or Nvidia GPU. It is primarily developed by Montreal Institute for Learning Algorithms (MILA [70]) and contains contributions from the open source community. A detailed tutorial on deep learning algorithms [71] covers the necessary steps in order to implement them in Theano.
Theano helps with the definition of mathematical expressions based on tensors and their evaluation on parallel architectures (either CPUs of GPUs). Since Python code is used for the computational graph definition, each run involves the automatic production and compilation of optimised C code that is responsible of the effective computation. A number of different approaches can be exploited for the configuration of underlying computational libraries, a permanent way to do this is through the definition of a configuration file in a home folder.
A number of Python packages can be used to speed-up Python code development by providing an high level API with a number of deep learning layers, Lasagne [72], Keras [73].
Unit tests are available (in source tree) for checking libraries and hardware access of current install; Lasagne example based on MNIST classification. Benchmarks not directly provided by the project, but the SDK is considered in [35].
Intel forked Theano and released an optimised version [74] tailored for Intel Xeon and Xeon Phi processors that is able to transparently improve performances on CPUs.

7.4.2.5 Tensorflow

TensorFlow [43] is an open-source framework for Machine learning. It is a system for building and training neural networks to detect and decipher patterns and correlations, analogous to (but not the same as) human learning and reasoning.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open source license on 9 November 2015. TensorFlow is Google Brain's second generation system. Tensorflow is today one of the most used deep learning frameworks. Today, the majority of Google products (Gmail, Google Photos, Voice recognition...) are based on this SDK.
It eases the definition, optimization, and efficient evaluation of mathematical expressions involving multi-dimensional arrays (tensors).
This SDK offers building blocks for the implementation of deep neural networks and machine learning techniques and transparently manages CPU and GPU computing resources. Moreover, it provides high scalability of computation across machines and huge data sets.
Project web page contains tutorials and examples [75], along with API documentation [76], web page contains benchmark results for recent GPU hardware [77].

7.4.2.6 Keras

Keras [73] is an open-source Deep Learning library for Python. The main idea of Keras is to provide a software layer on top of Deep Learning frameworks such as Tensorflow, Theano or CNTK. This abstract layer reduces development time when applying on different frameworks. Keras can run seamlessly on CPU and GPU.
Numerous implementations of commonly used neural network building blocks such as layers, objectives, activation functions, optimisers are offered, and a host of tools to make working with image and text data easier. Official website includes the documentation of the APIs and a collection of tutorials. Github [78] includes community-contributed projects and models.

Tool	First Release	Current Release	API	Supported HW/SW	Packaging	Prerequisites
Spark	1.0 (May 2014)	2.2.0 (July 2017)	Java, Scala, Python, R		Pre-built tar files (with/without Hadoop 2.6/2.7); source code.	Java, Scala
IBM Spectrum Scale (GPFS) support for Hadoop	2.7.0 (Dec 2015)	2.7.3 (June 2017)	HDFS compliant shell command	x86_64 (RHEL, SLES, Ubuntu), ppc64 (RHEL), ppc64le (RHEL, SLES, Ubuntu), Hadoop Distros (Apache, Hortonworks Data Platform, IBM BigInsights IOP)	rpm and deb packages.	Spectrum Scale (recommended 4.1.1+), Hadoop (2.7); minimum hardware requirements of 1 CPU (processor core) and 4GB to 8GB physical memory on each node
RDMA for Apache Spark (HiBD)	0.9.1 (Jan 2016) for Spark, 0.9.0 (Jun 2013) for Hadoop	0.9.4 (Mar 2017)		InfiniBand, RoCE, and IPoIB. It also provides support to run MapReduce jobs on top of Lustre alone. x86_64 architectures. (it should be available for openpower platform very soon)	library file dynamically loaded by Spark.	x86_64 architecture, infiniband network, OFED software.

Table 11: Apache Spark and Spark connectors for HPC.

Tool	First Release	Current Release	API	Supported HW	Packaging	Prerequisites
Torch	Oct 2012	V7 (Feb 2017)	Lua (primary, for users) C/CUDA (secondary, for developers).	Nvidia GPUs (multiple GPUs per node). Community-provided packages offer limited support to OpenCL, multithreading, MPI, MapReduce, and distributed computing through async and IPCs.	Complete package with Lua (language), LuaJIT (compiler), and LuaRocks (package manager).	CUDA
Intel DAAL	April 2016	DAAL 2018 Beta Update 1 Jun 13 2017	C++, Java, Python	x86_64, improved optimizations for newer Intel® processors, including Intel® Xeon Phi™ Processors 7xxx (codenamed Knight's Landing) and Intel® Xeon® Processors E5-xxxx v4 (codenamed Broadwell)	Source code [64], precompiled binaries or Anaconda package (Intel channel)	One of the following validated compilers: Intel(R) C++ Compiler for Linux OS 16.0/17.0, GNU Compiler Collection 5.1 and later, Sun Microsystems Java SE 8
Caffe	2014	1.0 (April 2017)	C++, Matlab and Python	x86_64, Nvidia GPUs	Sources on github, BVLC maintained Docker image in Docker hub.	CUDA, BLAS primitives (ATLAS, MKL or OpenBLAS), Boost, protobuf, glog, gflags, hdf5.
Theano	August 2011	0.10.0beta2 (Sept. 2017)	Python	x86_64, Nvidia GPUs	Sources on github, both CPU-only and CUDA-enabled Docker images on Docker hub.	Python, NumPy, SciPy, BLAS primitives (MKL, OpenBLAS), CUDA, libgpuarray, pycuda skuda.
Tensorflow	November 2015	V1.3.0 (Aug. 2017)	Python, C++, Java, Go	x86_64, Nvidia GPUs (multiple GPUs per node)	pip, sources on github	CUDA toolkit 8.0 or greater, cuDNN v3 or greater, GPU card with CUDA Compute Capability 3.0 or higher
Keras	27 March 2015	V2.0.8 (25 August 2017)	Python	Nvidia GPUs (multiple GPUs per node). Distributed computing with spark technology is available with third party components such as elephas [79]	pip, sources on github	Python numpy, scipy, six, pyyaml, CUDA toolkit 8.0 or greater, cuDNN v3 or greater, GPU card with CUDA Compute Capability 3.0 or higher

Table 12: SDK for Deep Learning on HPC systems.

8 Conclusions

In this deliverable, we have presented the work done within Task 6.2 of Work package 6 of PRACE-5IP project in period January–September 2017. We considered four services that have started already in PRACE-4IP and need further testing or transformation to regular service:

Service 1: The provision of urgent computing
Service 2: Links to large-scale scientific instruments
Service 3: Smart post processing tools including in situ visualisation
Service 4: Provision of repositories for European open source scientific libraries and Applications

Additionally, we consider two new services:

Service 5: Evaluation of lightweight virtualisation technology
Service 6: Evaluation of new prototypes for Data Analytics services

The work in Task 6.2 was coordinated by ULFME. Partners working on this task were grouped in six subgroups, one for each service.

For each service, we presented state-of-the-art in the area related to the service, the work done by the end of September 2017 and precise implementation plan till the end of the project. For Service 4 we decided to migrate into regular service, while for Service 1 we still consider how to proceed with it.

We paid special focus to pilot implementations of the services. The services are quite different and the same is true for the pilots. But it important that all the pilots will be developed and tested in cooperation with real back-end users and all of them will run on HPC.

The experiences with the pilots and proposals for the future development of the services will be included in the deliverable D6.4 that is due at the end of the project.