D7.1

First report on the activity of High-Level Support services (first year)

Mariella Ippolito, Fabio Affinito, and Giovanni Pizzi

Due date of deliverable:	30/11/2019
Actual submission date:	29/11/2019
Lead beneficiary:	CINECA (participant number 8)
Dissemination level:	PU - Public

Document information

Project acronym:	MAX
Project full title:	Materials Design at the Exascale
Research Action Project type:	European Centre of Excellence in materials modelling, simulations and design
EC Grant agreement no.:	824143
Project starting / end date:	01/12/2018 (month 1) / 30/11/2021 (month 36)
Website:	www.max-centre.eu
Deliverable No.:	D7.1

Authors: Mariella Ippolito, Fabio Affinito, and Giovanni Pizzi

To be cited as: Ippolito et al. (2019): First report on the activity of High-Level Support services (first year). Deliverable D7.1 of the H2020 project MAX (final version as of 29/11/2019). EC grant agreement no: 824143, CINECA, Bologna, Italy.

Disclaimer:

This document’s contents are not intended to replace consultation of any applicable legal sources or the necessary advice of a legal expert, where appropriate. All information in this document is provided "as is" and no guarantee or warranty is given that the information is fit for any particular purpose. The user, therefore, uses the information at its sole risk and liability. For the avoidance of all doubts, the European Commission has no liability in respect of this document, which is merely representing the authors' view.

Content

1. Executive Summary

The MAX outreach activities are largely based on offering services of interest to a broad range of users, both academic and industrial.
We guarantee high level support services to use MAX codes in the materials science area in general. This support is offered in two different ways: on one side there is the support offered by each code community *via* forum/mailing lists, where the responsible is always a specialist acting on behalf of MAX; on the other side there is the support explicitly operated *via* the MAX Help-desk. This offers both an advanced support as well as a consultation for the utilisation of codes.

MAX support activities aim also to produce a container-based deployment strategies to allow for easy access to codes and workflows of the MAX ecosystem.

2. Code and domain-specific support

As already mentioned, the code and domain specific support is operated on one side by the code communities, *via* forums/mailing lists, with specialists acting on behalf of MAX and on the other side *via* MAX Help-desk service. The support activities carried out by the communities of the individual codes and the ones operated by the MAX Help-desk will be reported here separately.

Since for each MAX code there is a support community already well established, we have evaluated that it would not have been appropriate to eliminate the already existing support communities of the single codes, in order to centralize all the support in a single point. Rather, we have tried to rethink and reorganize the support activities for each code so that these activities are always coordinated by the MAX staff. To emphasize and to advertise that this support activity is carried out within MAX, we have established that each support email would have been branded with a MAX signature and/or the MAX logo in the header of the web page dedicated to a MAX code support.

In order to complement the support already offered by the code communities and the one offered by the computing centres and PRACE, we create a MAX Help-desk that supports the users for all the codes in MAX area (in addition to MAX codes, we offer support for example also for Vasp, LAMMPS, CPMD, ADF, ...).

In the first phase of MAX, we offered an Help-desk service, creating a TTS system (Trouble Ticketing System) for MAX with the aim to have a central point for the support requests of all the MAX flagship codes. In order to send a request, the users needed to register on the MAX User Portal and then a second registration was needed on the MAX TTSsystem. The user's requests could only be sent through the User Portal and for each of them a ticket was opened. Afterwards, the ticket was managed by MAX staff until its resolution, involving - in case of need - the support by staff of the computing centres of the code developers. This procedure was proved to be impractical, with the result that only few requests for MAX support were sent through the MAX User Portal, while the support requests were sent mainly to the support team of the individual codes or directly to the CINECA TTS. In light of this we committed to organize MAX Help-desk differently.

In order to make easier and straightforward for a user to send a support request, we create the email address *support@max-centre.eu*. The mail messages directed to this address are forwarded automatically to the Help-desk service operated through the CINECA TTS system where, for each request, a ticket is opened in a queue dedicated to MAX. Also, the requests for codes belonging to MAX area which are sent directly to the CINECA User Support team are classified in the MAX queue and taken in charge by the MAX staff. Depending on the level of desired support, the requests are addressed by MAX staff or when necessary they take over by other pertaining support services. Then they are tracked until the resolution of the problem.

Since October 18th, 2019 the new MAX website is online and in the Help-desk section[1] it is specified how to ask MAX support and what kind of service is guaranteed.

2.1 Support activities

Typical domain-specific support actions can be distinguished in:

- investigation of the possible causes of a job failure and solution of the related problems. Typically this requires to re-build the code using different optimization parameters, or linking different libraries. In easy cases, it is only necessary to correct some input parameters or specifications of the computational resources in the batch script;
- evaluations of the MAX codes performances on different architectures. For example, this is necessary for selecting the best architecture to be requested in a PRACE access proposal;
- analysis of the MAX codes that behaves differently from documentation (e.g. non-converging algorithms);
- debugging of problems due to a specific code implementation (e.g. GPU, MPI, OpenMP versions);
- selection of the best code parameters that minimize the time to achieve a converged solution (e.g. input parameters or run time parameters);
- optimization of the computational setup to maximize the performances of a certain simulation on a given architecture (e.g. find the best combination of number of nodes, number of cores, number of mpi processes/threads, etc...)
- support in the usage of different releases of MAX codes (user guidance about new *vs* deprecated features).

[1] http://www.max-centre.eu/services/max-help-desk

2.2 Report on the first level support activities and KPI breakdown

The Key Performance Indicator (KPI) for this WP is given by the total number of support requests addressed by the CoE (trouble tickets, consultancy). In this regard, we collect the data related to the support activities operated by each MAX code community (Table 1) and the ones related to the MAX Help-desk activities (Table 2).

Since the support activity is organized in a different way for each MAX code (e.g forum, mailing list, Slack messages, ...), in Table 1 we report several columns to distinguish the different ways in which support was requested/provided. The "incoming threads" column represents the number of different requests arrived *via* email or in a forum. However, the total number of threads is not representative of the support activity carried out because each thread is typically followed by other emails/posts until the problem is solved. In order to take this into account, we report in the third column the total number of emails/posts for each MAX code.

Beyond the support offered through the official channels, there are support activities provided with several methods.

- The "face-to-face" support refers to an assistance provided in a more personal and direct way. Rather than relying on email, the "face-to-face" service is given by a personal interaction, e.g. *via* telephone, *via* skype call or being physically present in the same location.
- The "Gitlab threads" are the issues opened on the gitlab space of a given MAX code. Here we consider only the requests managed by MAX personnel. To use gitlab is necessary to be registered and the subscription is free.
- The "Slack messages" column contains the total number of messages shared in the Slack channel. Slack is a collaboration hub where conversations take place through channels organized by topic, project, team, or any issue considered relevant. In order to use Slack, it is necessary to be registered and also in this case the registration is free. As can be seen from the table, both AiiDA and BigDFT use intensively Slack as a way to support their users.

MAX Codes	Number of incoming threads	Total number of emails/posts	Gitlab threads	Number of face-to-face support activities	Number of slack messages
QE	272	386	30	-	-
Siesta	147	397	26	-	-
Yambo	161	394	-	9	-
BigDFT	34	48	45	11	15109

Fleur	46	167	60	2	-
AiiDA	84	323	909	4	4332
Total	738	1711	1070	26	19441

Table 1. Report M1-12 of the support operated by the single code communities and by MAX personnel.

The support activity operated by the MAX Help-desk is reported in Table 2, and in this service it is included also the support given for non-MAX codes largely used in the material science area. We report explicitly the support activity for Vasp and LAMMPS, that are the most used software, while we gather in “Others” label the support given for all the other codes (e.g. ADF, CPMD, Abinit). In this Table only two columns are present, one referring to the number of different threads and the other referring to the total number of received emails for each code. As already mentioned, each thread corresponds to a ticket opened in the MAX queue at CINECA system. Referring to Table 2, about 80% of the requests come from Italy, the rest from other European countries (mainly France, Switzerland, Germany). A statistics on the origin of the support requests managed by the communities of the various MAX codes is not available: however, we can say that in this case the support requests come mainly from Europe and North America, while a smaller percentage comes from countries like Asia and South America.

In M1-12 period covered by this data collection, there have been 98 service support actions by the Help-desk for MAX flagship codes and 293 related emails. Additionally, there have been 112 service support actions directed to other codes and 266 related emails, for a total of 210 actions and 559 emails. There were no support requests for AIIDA and Fleur, and only one request for BigDFT. Users of these three codes refer mainly to support provided by the codes community.

Codes	Number of Incoming threads	Total number of emails
QE	50	145
Siesta	23	83
Yambo	10	31
BigDFT	1	2
CP2K	14	32
Vasp	73	160
LAMMPS	20	54

Others	19	52
Total	210	559

Table 2. Report M1-12 of the support operated by MAX Help-desk. The row "Others" refers to non-MAX codes in addition to Vasp and Lammps, such as ADF, CPMD, ABINIT and CRYSTAL.

3. High level consultancy in materials science

The high level support targets more complex problems, typically taking longer time to be solved and requiring to develop *ad-hoc* solutions for the customer, involving a code development or a refactoring.

So far, the requests for high level support have been sent directly to the code communities support of each MAX code or to the MAX Help-desk. Within the new MAX website, it is now possible to send such requests directly through the "High level consultancy" subsection in the "Services" section, by filling a dedicated form[2]. This procedure is simple and immediate to submit and requires no registration.

All the high level support activities performed for the MAX codes in the first 12 months of the project are summarized in Table 3. Most of the requests have been already solved, only few of them are still pending.

The received requests for specialistic support are too many to comment on each of them. Giving few general example: concerning QE, most support activities refer to *ad-hoc* modifications in the code or in its post processing tools; referring to BigDFT, the majority of interventions are done for issues in using python interface PyBigDFT; instead, regarding Siesta, the ICN2 group is offering high-level support for several NFFA projects ad HPC-Europa projects, new features and new analysis tools were developed.

A large effort was devoted in performing CP2K benchmarks on different architectures, i.e. Intel knights landing (knl) nodes, Intel Skylake (skl) nodes, and Intel broadwell (bdw) nodes equipped with nVidia K80 GPU. Strong scaling and weak scaling curves were obtained for different case studies. These benchmarks were performed to support the user in choosing the most suitable architecture to be requested in PRACE proposal under preparation. The best performances were obtained on Intel Skylake (skl) cluster. Presently the PRACE proposal has been submitted.

Another relevant activity is the support given to an industry, AirLiquide[3], for running high-pressure calculation with QE. Their runs were failing on Intel Skylake nodes, whereas the same calculations were running without any issues on different architectures (e.g. knl

[2] http://www.max-centre.eu/services/max-high-level-consultancy

[3] https://www.airliquide.com/industry

nodes). We identified a bug in QE 6.4 related only to ScaLAPACK library. The bug was solved and we supported the user in finding the best set up for high-pressure calculation with both pw and cp. After the problem solution and a positive feedback from the industry, a possible collaboration and continuous support is currently under evaluation.

Finally, we describe here a significant activity not reported in Table 3 and related to PWQMMM code. Such software interfaces LAMMPS with QE to perform QM/MM calculations for both mechanical and electrostatic coupling. This code was developed in the first phase of MAX and the interface worked well until QE 6.2. Many relevant modifications were performed in QE from version 6.3 onwards and the outcome is that the interface with LAMMPS does not currently work. The organization of many libraries changed (for example LaXlib, UtilXlib and FFTXlib were introduced) and this required to link different libraries to build PWQMMM than it was before. Also, the organization of the MPI communicators changed and this fact introduced a bug causing the failure of the PWQMMM jobs because of a bad definition of some communicators used to exchange data between the QE and LAMMPS. A lot of effort was required to identify the bug, but it was solved so that currently the code PWQMMM works as expected.

Owner of the request	Applicant	Code	Date of request	Status	Type of request
M. Ippolito (CINECA)	Ivan Girotto (ICTP)	CP2K	08/04/2019	Closed	CP2K benchmarks on different architectures for a PRACE Proposal
P. Giannozzi (IOM-CNR)	Giovanni La Penna (CNR)	QE	03/2019	Closed	Parallelization of QE post processing code, under development, for the calculation of delocalization indices
P. Giannozzi (IOM-CNR)	Marco Pala (CNRS)	QE	04/2019	Closed	Interfacing QE data to custom transport code under development
P. Giannozzi (IOM-CNR)	Friedrich Prinz (Stanford)	QE	29/04/2019	Closed	Kinetic energy printout in QE
P. Giannozzi (IOM-CNR)	Alessandro Fortunelli (ICCOM)	QE	01/10/2019	Open	Additional term in forces for MD
M. Ippolito (CINECA)	Federico Iori (AirLiquide)	QE	16/05/2019	Closed	Industrial support for running high-pressure calculation with QE on Intel Skylake nodes (identified a bug in qe/6.4

					related to ScaLAPACKlibs)
Leopold Talirz (EPFL)	Iurii Timrov (EPFL)	AiiDA	15/05/2019	Closed	Support for deployment of custom Quantum Mobile for the tutorial "Eastern Africa School on Electronic Structure Methods and Applications: electrochemical energy storage and conversion materials"
Elsa Passaro (EPFL)	Chris Pickard (University of Cambridge)	Materials Cloud	01/2019	Open	Graphical maps of materials, a tool to graphically view material structures on the Materials Cloud. It is not public yet.
Sebastiaan Huber (EPFL)	Espen Flage-Larsen (University of Oslo)	AiiDA	11/12/2018	Closed	Change the heuristics of the `AIIDA_PATH` environment variable
L. Genovese (CEA Grenoble)	G. Jomard (CEA Cadarache)	BigDFT	02/2019	Closed	Usage of PyBigDFT interface
D. Caliste (CEA Grenoble)	M. D'Alessandro (CNR)	BigDFT	03/2019	Closed	Python3 issue with BigDFT compilation
L. Genovese (CEA Grenoble)	M. Stella (Imperial College London)	BigDFT	02/2019	Closed	TDDFT module for spin-polarized molecules in PyBigDFT
L. Genovese	A. Rago (Plymouth Univ.)	BigDFT	03/2019	Closed	Usage of SystemCalculator in PyBigDFT
A. Marini	M. Ferri (SISSA Trieste)	Yambo	09/2019	Closed	Development and fixing of self consistent calculations with Yambo
D. Varsano	S. Dong (Beijing Institute of Technology,China)	Yambo	10/2019	Closed	Usage of Yambo for BSE calculation on top of hybrid (HSE) ground state functional.
D. Sangalli	F. Affinito	Yambo	10/2019	Closed	Compilation of Yambo on

	(Cineca)				Power9 nodes of DAVIDE CINECA machine
A. Garcia (CSIC)	B. Hackens (Universite' catholique de Louvain)	SIESTA	06/2019	Closed	Compilation of SIESTA PEXSI on MareNostrum 4
S. Achilli (ICN2)	C. Meyer (University of Osnabrueck)	SIESTA	06/2019	Closed	Implementation of the Zeeman field in SIESTA
A. Garcia (CSIC)	M. Garcia Mota (SIMUNE)	SIESTA	09/2019	Open	Development of analysis tools for non-collinear spin calculations in Siesta, including spin-orbit coupling.

Table 3. Report M1-12 of the high level support activities.

4. Container technologies on HPC systems

The goal of this task is to produce container-based deployment strategies to allow for easy access to codes and workflows of the MAX ecosystem.

As a first step, we create a working group formed by at least one representative *per* MAX flagship code and/or *per* supercomputer centre. Within the working group, we evaluate the different use-cases covered within MAX.

4.1 Major use-cases for container technologies in MAX

We identify two major use-cases:

1. use of container technologies to facilitate optimal usage of *massively-parallel HPC codes on HPC resources;*
2. use of container technologies to ease *installation and deployment of HTC tools* (AiiDA, AiiDA lab)

For the two use-cases outlined above, we then determine the best container technologies as discussed below.

Use-case 1: performant use of HPC codes

For use-case 1 (HPC codes), we have identified two main technologies that are available in the HPC centres: **Shifter** (https://github.com/NERSC/shifter) and **Singularity** (https://www.sylabs.io/docs/).

Both technologies allow to define images and containers with the aim of providing efficient use in a parallel (MPI or other) environment, with possibility to use advanced hardware and accelerators (like GPUs) from within the container. Both, therefore, aim at overcoming the limitations that Docker poses in a HPC environment. The model of the two technologies is different in terms of security/permission model and process namespacing model, so the choice between the two is eventually driven by technology availability on the supercomputer and by performance considerations.

In order to address the issue of having both technologies (and considering that each supercomputer centre typically offers only one of the two), we identify approaches to build both types of images starting from the same docker image, as well as to convert one image into another. Moreover, we investigate techniques to use a single language to define the images in any format, and then automatically build optimized images for all technologies of interest (like HPC container maker by Nvidia, HPCCM - see below). Furthermore, we carefully address the integration with GPU-acceleration libraries like CUDA/OpenCL.

As a first flagship code to address containerisation, we have chosen the BigDFT code. In particular, we have devoted particular care in understanding how to integrate code compiled with licensed compilers and libraries (MVAPICH, OpenMPI, Intel MPI, Intel MKL), with the constraint of including in the image only the parts that are allowed by the license.

To achieve this goal, two image versions have been implemented: **bigdft/sdk**, and **bigdft/runtime**.

The first version (**bigdft/sdk)** is a docker container that provides all dependencies and libraries needed to build the code with optimizations within a single container. In particular, it provides the Intel MKL libraries, the Intel Python distribution, the CUDA/OpenCL libraries, the MPI libraries, and a jupyter notebook server. Nvidia's HPC container maker (HPCCM) is used to script the recipe to build this container, allowing us to easily switch versions and MPI flavors (currently MVAPICH-GDR, OpenMPI with CUDA support, or IntelMPI without CUDA are all supported). Such a container is ideal for development and to build the runtime image.

The second image version (**bigdft/runtime**) is a stripped docker container with BigDFT compiled with CUDA/OpenCL, Intel MKL, and a runtime MPI library (available choices include MVAPICH, OpenMPI, and Intel MPI). This container is compatible with both HPC containerisation technologies (Singularity and Shifter) so that it can be used in any supercomputer centre being part of MAX, as well as in many more facilities around the world. **bigdft/runtime** container has the advantage of both being smaller and not containing compilers or other licensed components.

One of the performance optimizations implemented in this container is a technique enabling the efficient use of vector operations when available, without preventing the use of the container in architectures that do not have these instructions. To achieve this goal, the images include two versions of the code - with and without AVX2 vectorization. The appropriate library is selected by glibc at runtime, depending on the underlying processor capabilities. Additionally, seamless integration with the code is possible thanks to the jupyter server available directly inside the image.

All images are available on DockerHub, and the corresponding HPCCM recipes can be found on GitHub; links are provided in the section below.

The use-case of BigDFT allow us to investigate in depth all aspects needed for providing efficient and performant containers for massively-parallel codes. Similar efforts are now in progress to extend the coverage of container technologies to all other HPC codes that are part of MAX.

Use-case 2: deployment of HTC codes

The second investigated use-case is the adoption of container technologies for the deployment of HTC codes.

In this case, there is no need to run directly on the HPC machines, and therefore all issues related to multiple compilers and to efficient interaction with accelerated hardware (GPUs, …) are less impactful. It becomes relevant to resort the container technologies in the following use-cases:

- deploying the code with all its dependencies (database, message queue broker) in a new machine;
- deploying the code on a scalable architecture, like kubernetes (https://kubernetes.io/), to be served efficiently on the web (e.g. for the AiiDA lab jupyter-based platform, developed by WP5);
- installing, configuring and running the codes on the native architecture rather than within a container or a set of containers.

For the first two cases, the best technology is Docker. In fact, it is already widespread for the deployment of codes including also multiple services interacting together, and it is ready to be used by kubernetes. Additionally, docker-compose (https://github.com/docker/compose - now fully part of docker) can be used to deploy all the services at the same time, keeping each of them in a different container to follow the philosophy of having each docker container run a “micro-service”. As an example in the case of AiiDA, a typical docker-compose deployment might include a machine for the PostgreSQL database, one for RabbitMQ message queue broker, one for AiiDA itself. The storage is typically bind-mounted directly from the host machine for performance reasons, or it could potentially be running on a fourth container (e.g. if storage is provided *via* an object store service like Swift or some other S3-compatible service).

Different use-cases are possible, ranging from a single fully-contained image including all services (e.g. for testing purposes) to some cases preferring a fully scalable solution based on docker-compose and/or kubernetes. Therefore we have developed a “docker stack” for AiiDA, composed by three images.

The first and lowest-level image, **aiida-docker-base**, is a basic installation of the AiiDA python framework with dependencies and the code, but without database nor setup. The goal is to be used either as a base docker image, or as one of the images in a docker-compose setup.

The second image, **aiida-docker-stack**, builds on top of *aiida-docker-base*, extending the AiiDA installation with a PostgreSQL database and a RabbitMQ broker within the same container. This image is particularly useful to test AiiDA within a single container.

Finally, we build a third image for the AiiDA lab service (a jupyter-based interface to provide easy GUIs to launch new simulations with AiiDA or perform data analysis, a deliverable of WP5). This is called **aiidalab-docker-stack** and extend the AiiDA installation of *aiida-docker-stack* with all the codes, plugins and services that are needed to run the AiiDA lab, including the *appmode* jupyter plugin to create seamless and intuitive web interfaces, as well as the AiiDA plugins for the MAX codes.

All these images are available on DockerHub (see links in the section below) as well as the corresponding Dockerfiles, available on GitHub.

We mention briefly that the third case, not properly considered as a container technology, is still very useful for maximal performance, and most importantly to deliver fully-working and self-contained virtual machines with all codes and dependencies installed, e.g. for educational purposes in tutorials or for demos of one or multiple codes. For this goal, the technique of choice is the use of ansible recipes (https://www.ansible.com/), used to deliver the Quantum Mobile virtual machine, developed in WP5 (https://www.materialscloud.org/quantum-mobile).

The ansible recipes are similar to Dockerfiles or HPCCM recipes, but they are designed in order to be idempotent, allowing for continuous upgrades of servers while they are running. In this specific case, there is no need of putting the servers offline or rebuilding every time from scratch.

In particular, using ansible recipes, it is possible to deploy automatically a full VM with all services preinstalled without any user intervention. Moreover, thanks to the integration with Vagrant (https://www.vagrantup.com/) all these services can be automatically packaged inside a VirtualBox image ready for distribution.

The advantage over container technologies for this educational use-case is that students can directly focus on the code and on learning the scientific details of the simulations, rather than having to learn how to install or manage the container technologies.

More details on this use-case will be presented in the deliverables of WP5.

The next steps for this task will involve:

- drafting documentation that explains: how to prepare the images, which technology is more suitable for which use-case, how to optimize performances of the applications inside the containers, and how to run the containers in the various supercomputer centres of MAX;
- contribution in all general container images to public repositories and markets to allow for further specialization of containerized applications, in particular extending to all other MAX codes in addition to BigDFT and AiiDA (whose images are already available, see links below);
- run performance benchmarks for the MAX codes on the various supercomputer centres, and then compare the performance of codes natively built in the HPC production environments with those run from within a container (if/when possible, compare the performance of different container technologies among them). The goal is providing recipes that are specialized for TIER0/PRACE systems and ready to scale on exascale machines.

4.2 Useful links to available containers and brief description

bigdft/sdk

URL: https://hub.docker.com/r/bigdft/sdk

HPCCM recipe:
https://github.com/BigDFT-group/ContainerXP/blob/master/hpccm/hpccm_lsim-mpi.py

Technology: docker, HPCCM

Main use: development, building of the runtime image

bigdft/runtime

URL: https://hub.docker.com/r/bigdft/runtime
https://ngc.nvidia.com/catalog/containers/hpc:bigdft

HPCCM recipe:
https://github.com/BigDFT-group/ContainerXP/blob/master/hpccm/hpccm_lsim-bigdft.py

Technology: docker, HPCCM (Singularity, Shifter)

Main use: HPC production runs, deployment

aiida-docker-base

URL: https://hub.docker.com/r/aiidateam/aiida-docker-base

Dockerfile: https://github.com/aiidateam/aiida-docker-base

Technology: docker

Main use: deployment, automated testing

aiida-docker-stack

URL: https://hub.docker.com/r/aiidateam/aiida-docker-stack

Dockerfile: https://github.com/aiidateam/aiida-docker-stack

Technology: docker

Main use: deployment, automated testing

aiidalab-docker-stack

URL: https://hub.docker.com/r/aiidalab/aiidalab-docker-stack

Dockerfile: https://github.com/aiidalab/aiidalab-docker-stack

Technology: docker, ansible

Main use: deployment

Notes: This can be used to deploy the AiiDA lab, using Ansible, within a single server using the following ansible recipe: https://github.com/aiidalab/ansible-playbook-aiidalab-server, or in a kubernetes cluster.

5. Conclusions

The organization of the support activities is giving positive feedback, considering both the support provided by the communities of each MAX code and the one provided by the MAX Help-desk.

The MAX website has very recently been updated, with a brand new section “Services”. Now, the specialistic support can be required directly by filling a simple online form. Thanks to the collaboration with WP9 in Communication and Dissemination, we will advertise the new support channel in order to promote it as main procedure for users to send specialistic support requests.

Concerning container technologies, the heavy work done shows its outcomes through the container images of AiiDA and BigDFT, that are now available in public repositories. The next step will be to prepare and make available the container images for all the other MAX flagship codes.