D5.2: First report on the VECMA infrastructure

Due Date	15 December 2019
Delivery	08 December 2019
Submission of updated version	N/A
Lead Partner	PSNC
Dissemination Level	Public
Status	Draft
Approved	Executive Board
Version	V1.1

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 800925.

DOCUMENT INFO

Date and version number	Author	Comments
07.10.2019 v0.1	Bartosz Bosak	Table of contents
30.10.2019 v0.2	Tomasz Piontek, Onnie Luk	Infrastructure requirements
31.10.2019 v0.3	Piotr Kopta, Vytautas Jancauskas, Erwan Raffin, Nicolas Monnier	Computing resources
04.11.2019 v0.4	Bartosz Bosak	Infrastructure conventions
06.11.2019 v0.5	Piotr Kopta	Infrastructure services
08.11.2019 v0.6	Bartosz Bosak, Hamid Arabnejad	Supplementary services & access tools
11.11.2019 v0.7	Bartosz Bosak	Tutorial & Summary
12.11.2019 v0.8	Tomasz Piontek	Jupyter Notebook
18.11.2019 v0.9	Onnie Luk, Olivier Hoenen	Proofreading
19.12.2019 v1.0	Tomasz Piotnek	Final check and approval
30.11.2019 v1.1	Bartosz Bosak Tomasz Piontek	Addressed internal reviewers' comments
08.12.2019 v1.1	Peter Coveney	Final comments and approval

CONTRIBUTORS

- Bartosz Bosak, PSNC – main author
- Tomasz Piontek, PSNC - WP5 leader, author, supervision
- Piotr Kopta, PSNC - author, Pilot Jobs
- Vytautas Jancauskas, LRZ - author, proofreading
- Onnie Luk, IPP - author, proofreading
- Hamid Arabnejad, Brunel (UBRU) - author, FabSim3
- Erwan Raffin, Bull - author
- Nicolas Monnier - author

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.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS AND DEFINITIONS

Acronym	Definition
API	Application Programming Interface
HPC	High Performance Computing
HTC	High Throughput Computing
MMSF	Multiscale Modelling and Simulation Framework
PRACE	Partnership for Advanced Computing in Europe
QCG	Quality in Cloud and Grid
VECMAtk	VECMA Toolkit
VVUQ	Validation, Verification and Uncertainty Quantification

1 Executive summary

This deliverable describes the status of the VECMA infrastructure in the 18th month of the project and it is the outcome of work realised within the work package "VECMA Infrastructure" (WP5). The VECMA infrastructure should be understood broadly as hardware and software systems offered to the VECMA project to enable and facilitate execution as well as verification, validation, and uncertainty quantification (VVUQ) of multiscale applications.

In the first 18 months, WP5 focused on activities that are required to instantiate the operational VECMA execution environment. Consequently, in accordance with the VECMA architecture developed [D5.1], the work package concentrated on the following three tasks:

- *Task 5.2 Creation and management of the project testbed and requirements to structure the work across different teams in VECMA*,
- *Task 5.4 Enabling middleware technologies for efficient execution of the VECMA simulations,*
- *Task 5.5 Development of intuitive user-level tools to facilitate the usage of the VECMA e-Infrastructure environment*.

The main duty of Task 5.2 is the construction of the project testbed consisting of several large-scale computing resources. At this moment, the testbed is fully functional and consists of two large HPC machines, namely Eagle and SuperMUC-NG, that are available for all VECMA partners, and a couple of smaller machines accessible only to specific institutions and communities. Additionally, Task 5.2 was responsible for the preparation of guidance where the common instruction of accessing VECMA resources as well as unified conventions and techniques required to homogenize the low-level execution environment are provided.

The role of Task 5.4 is to develop and provide infrastructure services. At this level, particular attention is given to the QCG execution services and to the QCG Pilot Job system, which are crucial for efficient multiscale application execution and VVUQ analysis.

The basic QCG execution infrastructure has been instantiated and made available to the VECMA community in the first year of the project. However, only the Eagle cluster was available at the time. Recently the infrastructure has been extended with the new SuperMUC-NG Tier-0 supercomputer, thus providing an opportunity to test distributed execution of VECMA scenarios.

In parallel, intensive work has been done to support execution of multiple tasks in many allocations (i.e., QCG Pilot Job mechanism) . Since the QCG Pilot Job system offers a solid basis such job executions and integrates easily with other tools in the VECMA Execution layer, we made it the primary Pilot Job mechanism to execute demanding UQ procedures in VECMA. Special focus was placed on its reliability and efficiency, as well as integration of the Pilot Job Manager with EasyVVUQ software (developed within WP3).

In addition, Task 5.4 initiated the work aimed to provide two new supplementary services to VECMA users: QCG-Monitoring and Jupyter Notebook. These services will enhance the infrastructure functionality, once they become available.

Last but not least, part of the work performed by WP5 in the first 18 months of the project was related to Task 5.5. The work done in this task concentrated on enabling seamless and flexible interaction with the infrastructure services by means of user-level access tools, particularly QCG-Client, FabSim3, QCG-Now and a client part of Jupyter Notebook. The efforts have been put to enhance the functionality of the tools as well as to adapt them to the circumstances of VECMA environment.

2 VECMA infrastructure

This deliverable describes the status of the VECMA infrastructure in the 18th month of the project and it is an outcome from the work done within Work Package "VECMA Infrastructure" (WP5). It is important to note that this document is a result of the cooperation of all project participants who, through the development of the VECMA toolkit (VECMAtk), [VECMAtk], directly or in-directly affected the form and actual characteristics of the infrastructure provided.

The current status of VECMA infrastructure is grounded to the requirements specified by the VECMA community. These requirements were collected in three phases: first, during the preparation of the project proposal; second, when the applications' developers responded to the infrastructure-related questions in the application questionnaire [D4.1]; and third, during the regular work between all technical VECMA work packages so far.

It is worth stressing that, in accordance with the assumptions made within the project, the term “infrastructure” may be interchanged with the more descriptive “execution layer”. In this sense, it refers to the widely understood hardware and software systems offered to the VECMA project in order to enable and facilitate execution as well as verification, validation, and uncertainty quantification (VVUQ) of multiscale applications. Therefore the VECMA infrastructure can be seen as a collection of integrated user-level tools, middleware services and distributed HPC/HTC machines.

The rest of this document presents the current status of the VECMA infrastructure. It begins with a short section summarising the collected requirements. The following sections describe the progress in implementing particular components of the VECMA execution environment as well as detail the additional actions undertaken by WP5. The document concludes with a section summarising the main achievements of WP5 and providing a short outlook for future plans.

2.1 Requirements for the VECMA infrastructure

The requirements of the VECMA project to perform VVUQ on complex, multi-scale applications place very high demands on the infrastructure. This becomes obvious when we analysed answers from the questionnaire given to all VECMA application developers and users [D4.1]. We also learned that one production run of a multiscale application may require as many as 100 000 cores reserved for more than one day. It is a challenging problem by itself. However, when we put it into the VECMA context and multiply that production run by thousands in order to perform reliable VVUQ, it shows a more precise picture of the problem. Our long-term objective in this project is to develop a generic system that allows external users to run third-party applications not only on existing peta-scale systems, but also on emerging exa-scale systems. And that makes the infrastructure implementation even more challenging.

Figure 1, which can be found in [D5.1] published at month 12, outlines the foreseen VECMA Execution (also known as Infrastructure) layer. In essence it consists of user-level tools, infrastructure services, of which the most important ones are execution and pilot job services, and the actual hardware resources. It would be extremely difficult to cover all of these elements without a portfolio of validated components.

Thankfully, as it was described in [D5.1], to some degree VECMA certainly benefits from previous EU projects. In the context of WP5, foundation of the infrastructure comes from the MAPPER project, where Multiscale Modelling and Simulation Framework (MMSF) was proposed [MAPPER project], and the ComPat project, where High Performance Multiscale Computing Patterns (HPMCP) were developed [ComPat project]. Within these projects several key VECMA software components, such as the core QCG services, were initially used to answer the collective needs of a substantial multiscale application portfolio.

A clear advantage at the initial stage of the project was the opportunity to utilise PSNC's Eagle cluster, which was also employed in ComPat. It is possible to reuse the previously-developed procedures for accessing this cluster. All we needed to do was to slightly adapt them to the specific needs of VECMA project in order to offer the cluster to VECMA users.

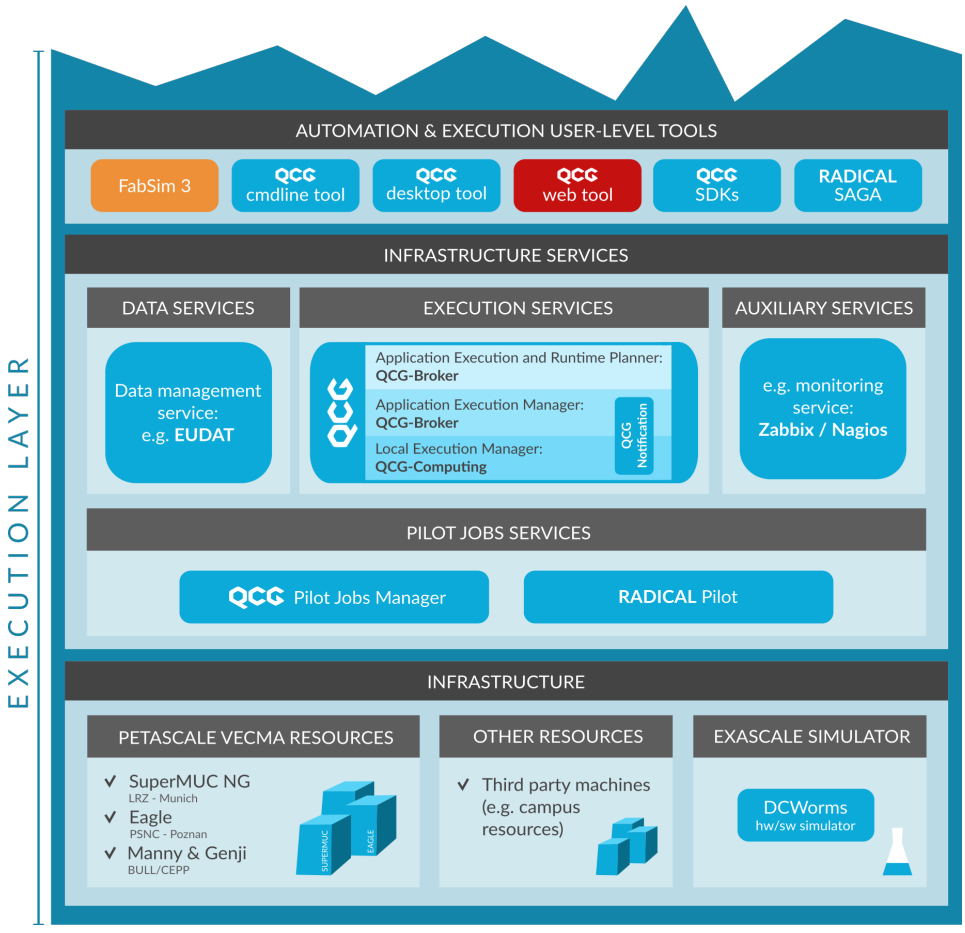

Figure 1 - Execution layer of VECMA architecture

With these elements in place, the VECMA application and toolkit developers could initiate their work in the project. However, several further requirements are not yet satisfied. One of which is of great importance to WP5: the efficient execution of VVUQ workflows that consist of a very large number of potentially-dependent computing tasks. We believe that this requirement can be fulfilled by extending the QCG Pilot Job mechanism to provide the possibility to distribute the execution of highly-demanding workflows on multiple computing resources.

In order to improve its uptake by other communities, special attention will also be given in developing user-level tools and providing new services that could facilitate daily usage of VECMA system once it is made production-ready. WP5, in collaboration with WP4 and WP3, have started defining the most desired extensions in this field. As a result, WP5 begins implementing extensions to existing set of VECMA components in order to address the need for an application level monitoring solution.

On the other hand, due to the limited interest from application providers to use the DCWorms simulator at this stage of their code development, a decision has been made (between the partners and the project coordinator) to move the efforts from Task 5.3 to develop the aforementioned monitoring solution, instead.

The report on the development of the infrastructure is the topic of the next section.

2.2 VECMA infrastructure development

The initial decision to split the VECMA development into two main phases (the release of fast-track components and the release of deep-track components) provides valuable benefits at this stage of the project. The release of the fast-track components occurred at month 12, thus it is now possible to clearly point out the backbone of the VECMA's infrastructure. In turn, the additional 6 months of work allows us to outline reasonable prediction for the final state of the system.

The rest of this section begins with a description of guidance created to provide basic information about the VECMA infrastructure to people involved in the project. Next, one by one, starting from the availability of computing resources, description of updates introduced to main infrastructure services, improvements to pilot job mechanism, and ending with the report about user-level tools and libraries, the progress of realisation of VECMA infrastructure in the 18th month of the project is summarised.

2.2.1 VECMA infrastructure guidance

For those who want to get detailed information on how to work within the proposed VECMA infrastructure, whether they are resource administrators, service provides, application teams or current/future VECMA system users, they can visit the infrastructure guidance wiki created by WP5, which is available here: [Infrastructure Wiki].

In order to simplify its usage to groups involved in or using VECMA, the guide is divided into four main sections, with each dedicated to a specific group, namely:

- **Information for end-users**
  Provides instructions targeted at all VECMA users, including VECMAtk developers, application

developers, as well as VECMA end-users. It gives extensive information about the VECMA infrastructure, how to access VECMA resources and how to start using them.

- **Information for application teams**
  Provides a basic guide for the VECMA application developers and/or maintainers. In particular it describes conventions and best practices for application deployment.
- **Information for people responsible for deploying VECMA tools on computing resources**
  Offers instructions for people responsible for deployment and maintenance of VECMA toolkit (VECMAtk) on resources. It provides guidance on how to deploy VECMAtk on resources in order to keep the infrastructure homogeneous.
- **Information for resource providers and administrators**
  Provides basic requirements and conventions of VECMA project that should be followed by resource providers and administrators in order to enable instantiation of the VECMA environment on the resources.

While the guide provided is a “living” set of instructions that could potentially change, the goal is to provide new instructions rather than alter existing ones. Therefore the existing content in the guide may be treated as a project oriented, dedicated reference for deployment, maintenance and use of VECMAtk and VECMA applications on the VECMA resources.

2.2.2 VECMA conventions

One of the priorities of the WP5 in the first phase of the project was to define rules that will ensure the consistency of VECMA environment, regardless of computing resources. However, the situation the work package wanted to avoid was the excessive complication of the procedures at this level of the system. To this end we have decided to provide only a few small regulations and store them as reference in the infrastructure guide [Infrastructure_Guide].

The set of elementary regulations that should be addressed in all VECMA computing resources and respected by all VECMA participants are as follows:

1 The computing resources should provide the support for environment modules.
2 There should be a top-level module for the VECMA project called 'vecma'.
3 A specific naming scheme should be applied to the modules, e.g. vecma/apps/app1 for a specific module to application ‘app1’ or vecma/common/lib1 for the library ‘lib1’ that may be useful for many applications.
4 The vecma module should define a common environment variable pointing to a shared directory on resources called "VECMA_DIR".
5 The structure of "VECMA_DIR" should correspond to the basic circumstances of VECMA project; it should include 'Apps' and 'Common' subdirectories for application-specific data and general-use data, respectively.

2.2.3 Status of VECMA computing resources

VECMA participants are allowed to select from a set of resources provided to the consortium by a subset of WP5 partners (PSNC, LRZ and Bull). The set includes two large supercomputers available to

the whole consortium, namely Eagle (PRACE Tier-1 class) at PSNC and SuperMuc-NG (PRACE Tier-0 class) at LRZ, and two smaller machines that are used internally for software testing purposes. More information about the core computing resources and allocations offered to VECMA by the project's resource providers is available in D5.1 [D5.1].

The following part of this section concentrates on the current progress in making the particular resources available and usable by the VECMA consortium.

Some of the project's partners use their own or external (to the project) resources to develop and test VECMA applications or particular VECMA components. Therefore, we will describe the role and the specifics of these third-party resources as well.

In general, resource availability in the context of VECMA can be considered at two different levels.

The resource is *available for the whole project* if the following criteria are met:

- accounts for all interested VECMA users are available,
- remote access to resources for all VECMA users is provided,
- shared disk space for the whole VECMA project is assured,
- the allocation of computing power (core hours) for the whole VECMA project is confirmed,
- key components of VECMA toolkit installed in a shared space for all VECMA users.

The resource can also be assumed as *available for the selected project members* if:

- accounts for selected VECMA users are available,
- allocation of computing power (core hours) for selected VECMA users is confirmed,
- parts of VECMA toolkit installed on user's space.

According to this categorisation, the aim of VECMA is to have two large computing clusters (namely Eagle and SuperMUC-NG) in the first group and a couple of variously-restricted clusters in the second group. The condensed view on the current status of availability to computing resources for the VECMA project is presented in Table 1.

Table 1 - Status of availability of VECMA computing resources

Resource	Status	Notes
Eagle (PSNC)	Available for the whole VECMA project in production mode.	Eagle is accessible with traditional ssh and SLURM interface, but also with QCG tools and FabSim3 Access instruction for new users is available here: [Eagle_Access]
SuperMUC-NG (LRZ)	Available for the whole VECMA project via accounts created on request	Due to unforeseen delays in installation of SuperMUC-NG at LRZ, the access to the system for external users has been delayed by several months.

		Currently the system is accessible via SSH and jobs are submitted using SLURM.
Manny (Bull)*	Available for the project participants from Bull	Used to be accessible with traditional ssh and SLURM interface. A new machine called Aloy is now replacing it.
Genji (Bull)*	Available for the project participants from Bull	Accessible with traditional ssh and SLURM interface
Marconi (Cineca)*	Available for the project participants from MPG and the Fusion community in Europe	Accessible with traditional ssh and SLURM interface
Cobra (MPG)*	Available for the project participants from MPG	Accessible with traditional ssh and SLURM interface
Draco (MPG)*	Available for the project participants from MPG	Accessible with traditional ssh and SLURM interface
ARCHER (EPCC)*	Available for the project participants from UBRU and UCL.	Accessible with traditional FabSim3+ssh and SLURM interface.
Cartesius*	Available to the project participants from UCL & UvA.	Accessible with traditional FabSim3+ssh and SLURM interface.
Hazel Hen (HLRS)*	Available for the project participants from UBRU.	Accessible with traditional ssh.
Prometheus (Cyfronet)*	Available for the project participants from PSNC	Accessible with traditional ssh and SLURM interface as well as with QCG tools
Tryton (TASK)*	Available for the project participants from PSNC	Accessible with traditional ssh and SLURM interface as well as with QCG tools

* the resource is available to selected project participants

As it can be recognized from the table, currently both Eagle and SuperMUC-NG are available for the VECMA users. Nevertheless, the transition period of replacing the old SuperMUC machine with the new SuperMUC-NG resulted in Eagle being the only publicly available cluster in VECMA for a fairly long time. This situation was certainly not optimal, but it didn't have much influence on the project implementation at the time. The strict rules undertaken in the project to develop portable software allowed VECMA to rely on the already-available resources - particularly on Eagle, but also on supplementary clusters such as Manny or Marconi. Thanks to the extensive utilisation of containerisation or PyPI repositories for VECMA software, the components developed in one place can be easily transferred to other locations. Currently SuperMUC-NG is already available for VECMA users. Presently only via SSH, but in the near future it will be configured to access from QCG and FabSim3 as

well. Once that is done both clusters will offer a homogenised infrastructure and a reliable testbed for demanding distributed execution of multiscale applications and VVUQ workflows.

Below we present detailed information about the current readiness status of internal VECMA resources as well as brief description of third-party resources known to be used by VECMA developers.

VECMA resources

Eagle

- User accounts for VECMA:
  Available for the whole consortium (realised via the PLGrid e-Infrastructure)
- Access to resources:
  ssh+SLURM, QCG clients, FabSim3
- Shared disk space:
  Yes - Lustre
- Core Hours budget:
  4M core hours for 2019, similar amount for the next years
- Applications running:
  Fusion, Climate, UrbanAir, FabFlee

SuperMuc-NG (LRZ)

- User accounts for VECMA:
  Accounts available for the whole consortium (via SSH)
- Access:
  ssh+SLURM
- Shared disk space:
  Yes
- Core Hours budget:
  Currently 100k CPU hours
- Applications running:
  FabFlee

Genji

- User accounts for VECMA:
  Accounts for the Bull partner only
- Access:
  Limited for the Bull partner: ssh+SLURM
- Shared disk space:
  Yes (for the Bull partner)
- Core Hours budget:
  At least 11281 core hours have been used for the project.
- Application running:
  The mini-apps Material and Migration.
  The cluster is used to test the VECMAtk and its scalability

Manny/Aloy

- User accounts for VECMA:
  Accounts for the Bull partner only
- Access:
  Limited for the Bull partner: ssh+SLURM
- Shared disk space:
  Yes (for the Bull partner)
- Application running:
  The cluster Manny was used to test the PJM and Fabsim3 scalability.
  Aloy replaces Manny.

Other resources

Marconi (CINECA)

- Traditional (SkyLake) and accelerator (KnightsLanding) for respectively ~8 and ~11 PFlop/s, Intel OmniPath
- Accounts for the MPG partner
- Allocation of 9.7M core hours for VECMA
- Application running: Fusion

Cobra (MPG)

- SkyLake with 11.4 PFlop/s peak DP + 2.64 PFlop/s peak SP, Intel OmniPath
- Accounts for the MPG partner
- Application running: Fusion

Draco (MPG)

- Haswell with peak performance of 1.12 PetaFlop/s, InfiniBand FDR14 network
- Accounts for the MPG partner
- Application running: Fusion

Prometheus (Cyfronet)

- Haswell with peak performance of 2.34 PetaFlop/s, InfiniBand FDR network
- Accounts for the PSNC partner via the PLGrid infrastructure
- Application running: UrbanAir

Tryton (TASK)

- Haswell with peak performance of 1.48 PetaFlop/s, InfiniBand FDR network
- Accounts for the PSNC partner via the PLGrid infrastructure
- Application running: UrbanAir

Carthesius

- Bullx system, Intel Xeon CPUs. Three main types of nodes Fat (32 cores), Thin (24 cores), GPU
- Accounts for the UCL partner
- Initial Allocation for VECMA.

ARCHER (EPCC)

- Cray XC30
- Accounts for the UBRU and UCL partners
- The initial allocation of 3.33k core hours (500 kAUs - ARCHER Allocation Units) for VECMA
- Applications running: FabFlee, FabSMD

Hazel Hen (HLRS)

- Cray XC40
- Accounts for UBRU, through the HiDALGO project
- Allocation of 10k core hours for VECMA
- Applications running: Flee, (FabFlee/FabSim and MUSCLE3 are yet to be connected to this platform).

2.2.4 Infrastructure services

A key objective of WP5 is to provide a unified and efficient access to the underlying infrastructure composed of distributed computing resources of various size, operational policies and local resource management systems. To achieve this, as it was described in D5.1, the work package relies on a well-established set of execution and Pilot Job services built upon the QCG solutions [QCG], which are supported by supplementary services like those for data management.

It should be noted that both clusters and deployed key services are monitored by the dedicated systems available in the institutions of origin. The PSNC's Eagle and QCG services are monitored by Zabbix and Nagios systems, respectively.

Execution services

There are ongoing deployment and maintenance activities to allow the VECMA consortium access to the project's resources via QCG execution services.

The current deployment status of QCG services is shown in Figure 2. In this diagram we can distinguish two sets of services: infrastructure-level services and resource-level services. The key service at the infrastructure level is QCG-Broker, which is the entry point to the whole QCG-managed environment. It is accompanied by an instance of QCG-Notification service. At the resource level, the key role is played by the QCG-Computing services, which need to be deployed on each site participating in the infrastructure. Detailed description of these components can be found in D5.1 [D5.1]. This deployment is complemented by gridFTP services available on resources to support data staging.

At this moment, the deployment includes common QCG-Broker and QCG-Notification services installed on a dedicated machine at PSNC, as well as one QCG-Computing service on Eagle. All of these services have been successfully integrated.

In the case of SuperMUC-NG there are still ongoing efforts to configure QCG-Computing according to local needs and integrate it with infrastructure-level services. Once SuperMUC-NG is fully functional

and stable, its access will be granted through QCG and it will be the largest computation resource for VECMA.

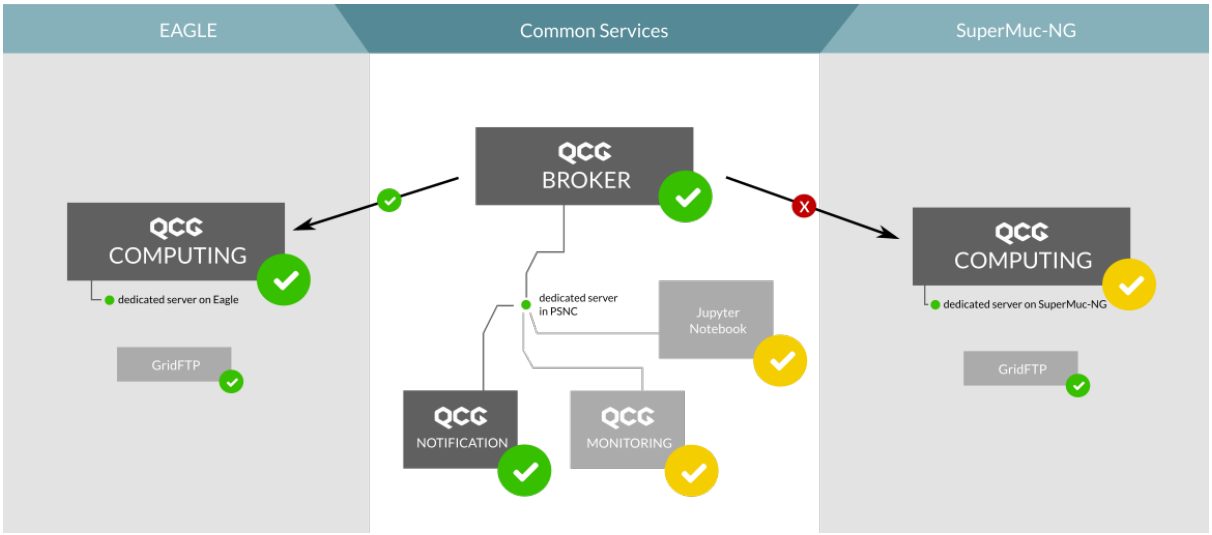

Figure 2 - QCG services deployment status

Pilot Job Services

There is a need in the VECMA project to execute a large number of computing processes. This is especially evident in the case of UQ procedures. This necessitates the use of a pilot job mechanism since it provides the required efficiency and flexibility. Further analysis, based on specific expectations of the VECMA research community, allowed us to identify a set of features that the Pilot Job system should provide. Based on this information, WP5 decided to concentrate on development of QCG Pilot Job system [QCGPJ] and its integration with VECMAtk as the main mechanism for execution of Pilot Jobs for both generic UQ procedures and application-specific execution logic. On the other hand, the implementation work to integrate Radical Cybertools Pilot Job mechanism with VECMA has been temporarily suspended. The continuation of this task will be resumed if there is interest from our users.

QCG Pilot Job

Execution and VVUQ analysis of multiscale applications are complex processes that shouldn't be further complicated by the purely technical aspects of the execution of computing tasks. From the researcher's perspective only the results of computations are important. These facts became a guideline in the development of QCG Pilot Job system. Thus the goal for both PSNC (as the main developer of this system) and WP5 (as its integrator in VECMA) is to provide a lightweight, highly flexible and easy to use solution - a solution that wouldn't interfere with the domain-specific aspects of VVUQ analysis and multiscale computing in general. Starting from month 10 of the project, as part of Task 5.4 of WP5, the QCG Pilot Job system has been improving continuously and adapting to particular needs of the VECMA project. Among the various types of advancements being introduced, the following should be noted in particular:

- Support for two modes of execution: local and inside a scheduling system allocation, for execution of Pilot Jobs on a personal computer and on an HPC/HTC machine, respectively. This allows one to seamlessly move the work to the targeted environment when basic functionality is already in place.

- Development of a new lightweight executor service to launch tasks on allocated computing nodes. This extension was motivated by poor performance of SLURM mechanisms in HTC scenarios, particularly the srun command that initially was used in the QCG Pilot Job system.
- Development of a hierarchical structure of Pilot Job Manager services in order to further improve scalability of the system.
- Design of an API that allows the instantiation of QCG Pilot Job Manager service directly from Python code. It simplifies the usage of Pilot Job Manager for Python developers as well as its integration with other Python-based tools, particularly with EasyVVUQ.

To ensure usability and scalability at HTC scenarios as required, the QCG Pilot Job Manager performance has been studied by both PSNC and Bull. Early study on the speedup (as a function of the number of compute nodes) to schedule 1000, 10000 and 20000 samples of a toy application (simple nBody running on 8 cores) with up to 7 Genji nodes (280 cores) showed that the scalability is relatively good regarding the theoretical Speedup (cf. Figure 3).

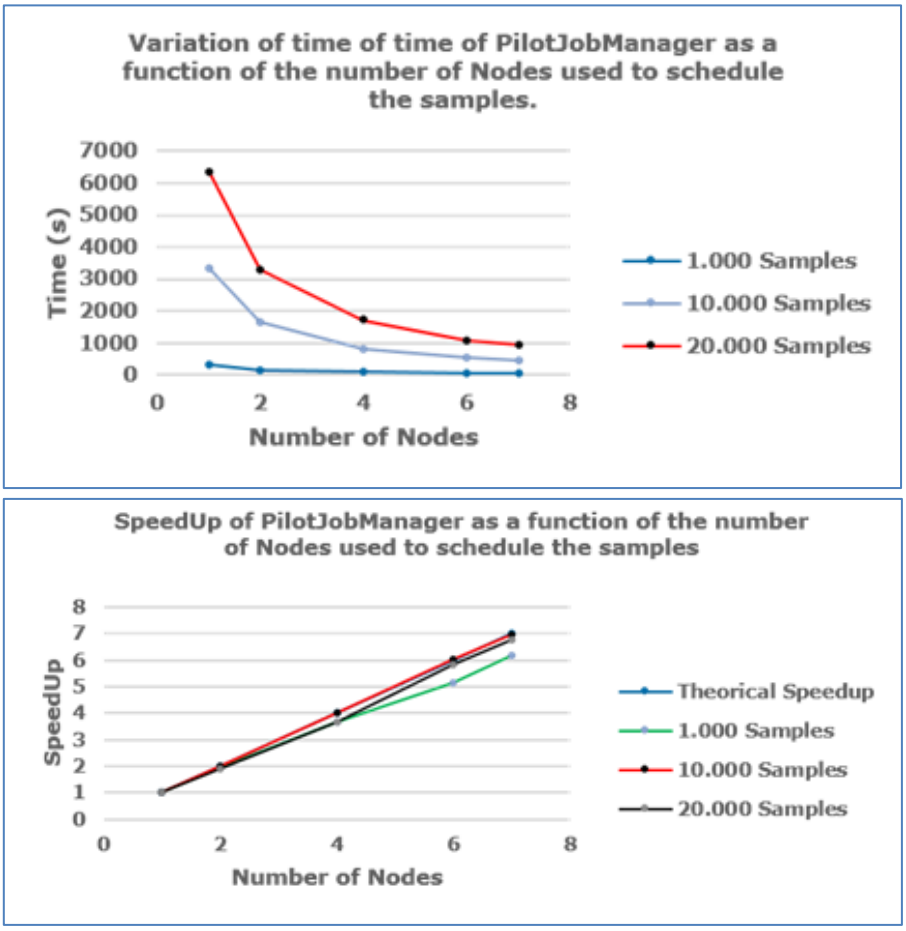

Figure 3 - Early scalability result of QCG Pilot Job Manager on Genji

Further tests with the real use-case i.e. the Fusion application led us to discover the bottleneck related to low performance of the tool in case of typical HTC scenarios, where a large number of short jobs is processed. In order to fix this problem, WP5 provided two improvements to the base solution. The first improvement was motivated by the discovered poor performance of the srun command used to start

Pilot Job tasks. The solution to this problem was the implementation of a dedicated lightweight executor service that replaced the standard SLURM mechanism. Although the scalability largely improved, there was still a bottleneck. This time, the problem was in handling the queue of tasks and detection of changes in the status of individual tasks. Since this problem was more generic and related to the current phase of the QCG Pilot Job development, the solution was in a natural and planned evolution of the tool. Currently there is an experimental version of QCG Pilot Job system resolving this issue already available. The introduced updates to the system concerned the support for the hierarchical structure of QCG Pilot Job Manager services. Previously, there was only one QCG Pilot Job Manager controlling the whole experiment, currently, there is a two level hierarchy: the QCG Pilot Global Manager service at the top level, and a number of subordinate QCG Pilot Job Partition Managers. This solution allows to effectively scale up Pilot Job instances to meet current and future application requirements. The solution concept itself can be replicated to the next levels of management, as a result creating an efficient mechanism for future exa-scale systems. The concept and architecture of the hierarchical QCG-PilotJob system is shown in Figure 4.

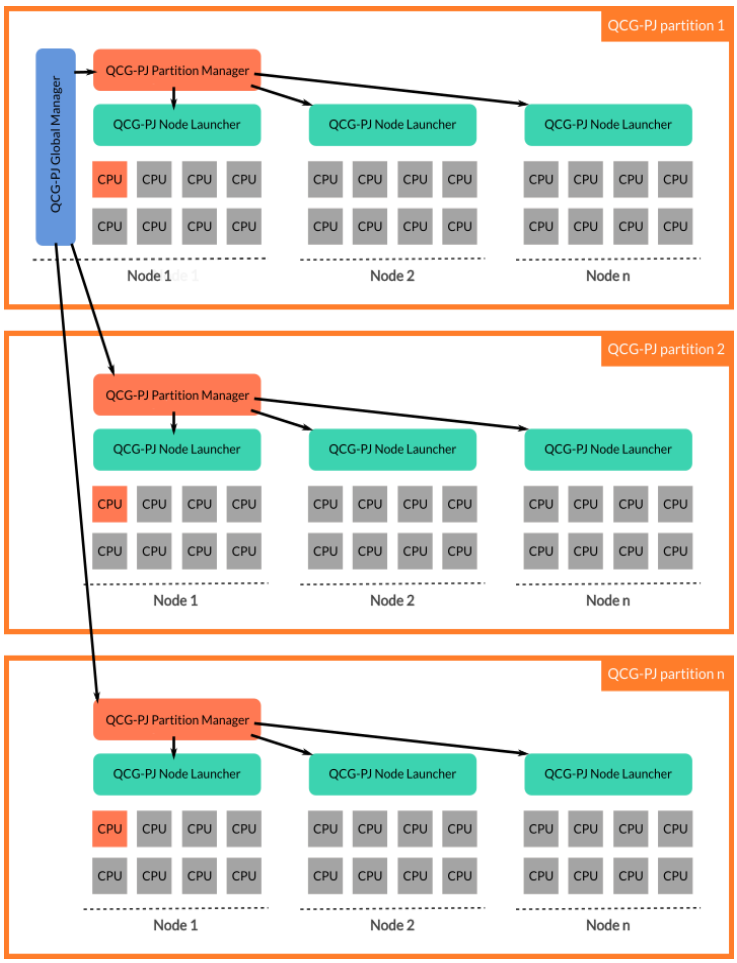

Figure 4 - Hierarchical structure of the latest QCG Pilot Job version. QCG Pilot Job Global Manager submits jobst to QCG Pilot Job Local Managers managing separate QCG Pilot Job Partitions. The orange color of one CPU per partition means that this core is reserved for QCG Pilot Job manager.

In order to show progress in HTC scalability of QCG Pilot Job achieved in the result of introduced improvements, the synthetic tests have been performed. Below, in Figure 5, there are shown graphs presenting the effectiveness of the QCG-PilotJob Manager service for the following configurations:

- one manager service controlling the whole allocation and starting tasks via srun queue system command,
- one manager service controlling the whole allocation and starting tasks with the use of its own service
- many manager services controlled by a global instance and starting tasks using its own service.

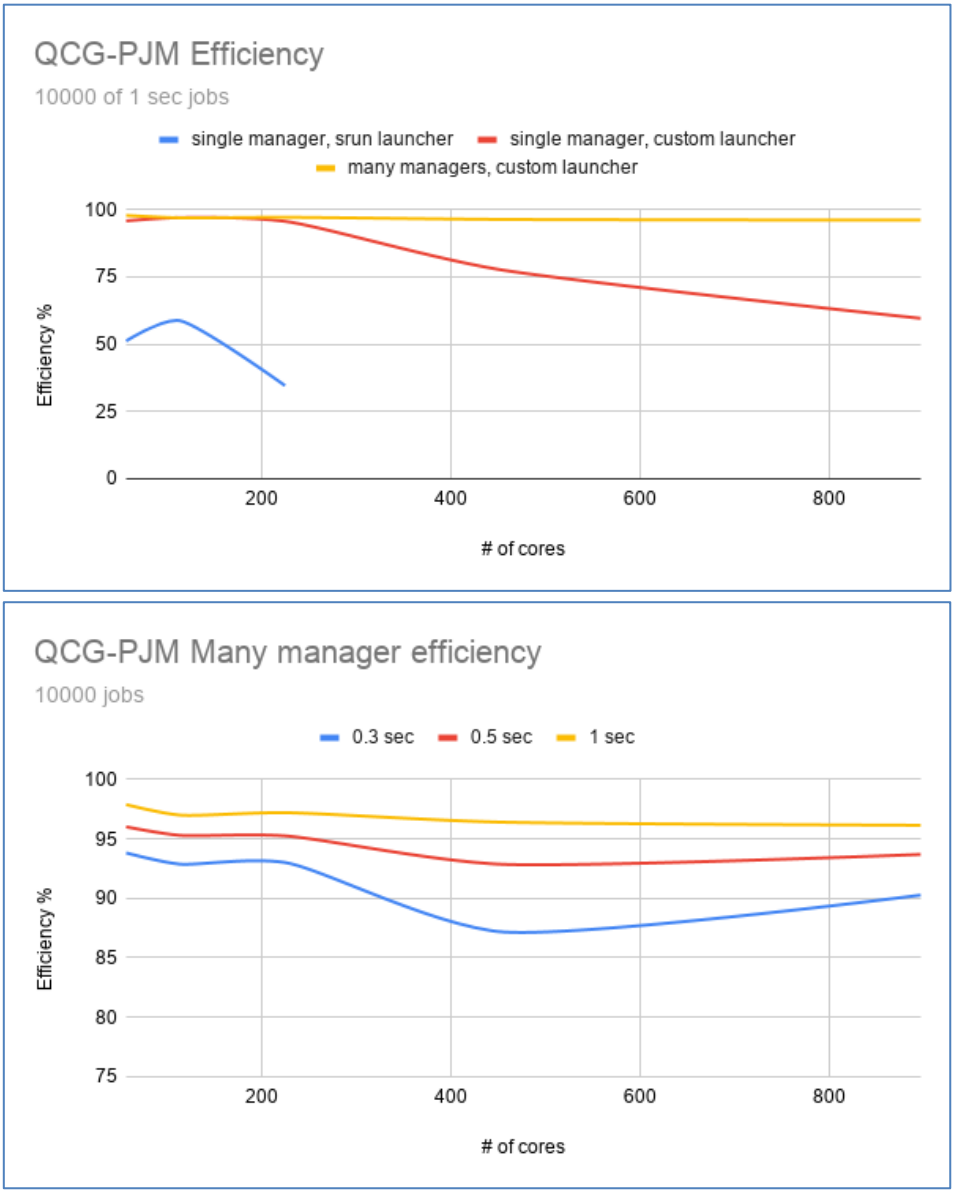

Figure 5 - Comparison of efficiency of different versions of QCG Pilot Job Manager

The efficiency of the service was calculated as a quotient of the theoretical and actual time of the scenario where the actual time did not include the time of service initialisation. The theoretical scenario execution time was calculated from the formula:

*CEIL(NUMBER OF TASKS / CORE NUMBER) * TASK COMPLETION TIME.*

The results show that thanks to the partitioning of available resources and the introduction of multiple instances of the manager's service, it has been possible to increase the system efficiency (up to about 97%), as well as its scalability (up to 3000 tasks per second). We plan further work related to efficiency testing through the use of larger instances as well as real applications.

Continuation of QCG-PilotJob throughput optimization is planned in parallel as part of a dedicated task in Package 3 (Task 3.4 Optimising scalability of the VECMA toolkit). The detailed description of the optimizations made as well as tests carried out will be included to deliverable D3.4 and D3.5.

EasyVVUQ-QCGPJ

In parallel to the development of QCG Pilot Job by WP5 and EasyVVUQ by WP3, the additional efforts are put to integrate these two tools and offer them to users as a boxed EasyVVUQ-QCGPJ product. The initial version of this implementation was released at month 12 of the project together with other VECMAtk components. It provides basic functionality required to execute encoding and execution phases of EasyVVUQ workflow as QCG Pilot Job tasks. The tool was next updated and adjusted to the various improvements introduced to EasyVVUQ and QCG Pilot Job at month 15 of the project. The current goal is to simplify the usage of the tool by hiding QCG Pilot Job execution details behind a simple-to-use API.

Supplementary services

Simultaneously with the actions related to maintenance of the core infrastructure services, the work on new services has been established. Based on the feedback received from VECMA partners, particularly alpha-users of the VECMA toolkit, we have decided to put additional efforts into adding two new components to VECMA, namely QCG-Monitoring and Jupyter Notebook environment. The purpose of these elements in VECMA is discussed below.

QCG-Monitoring

One of the basic requirements of scientists performing almost any kind of long-running simulation is the possibility to track the simulation's execution. The basic form of monitoring, which has already been embedded into the QCG middleware and available to all its users, allows one to receive email or XMPP notifications with excerpts from the application's output. Although this basic mechanism may be sufficient for simple applications, it is surely far from ideal for the more demanding scenarios, such as those targeted by VECMA. In that case, having a more descriptive and compact form of displaying the progress of execution, e.g. with tables or charts, is desired. Hence, VECMA imposes an additional requirement on monitoring solution to display the progress of execution at several distinct levels:

- the execution of single-scale application module,
- the execution of multi-scale application,
- the execution of replicas, and
- the execution of VVUQ workflow.

In order to satisfy the above VECMA community expectations, WP5 started to work on a monitoring system tailored to the VECMA use cases. The solution is based on the already-existing QCG-Monitoring service, which out-of-the-box offers to end users a customizable and flexible monitoring capabilities.

As it is shown in Figure 6, the QCG-Monitoring system consists of a central Monitoring Service and a number of monitoring schemes for specific applications. Each of these monitoring schemes includes a data extractor that is used to produce essential monitoring data from the output of a given application, and a display template, which is used to map the application output to a dynamically generated HTML webpage. The set of monitoring schemes is not limited and may easily be extended, depending on the requirements.

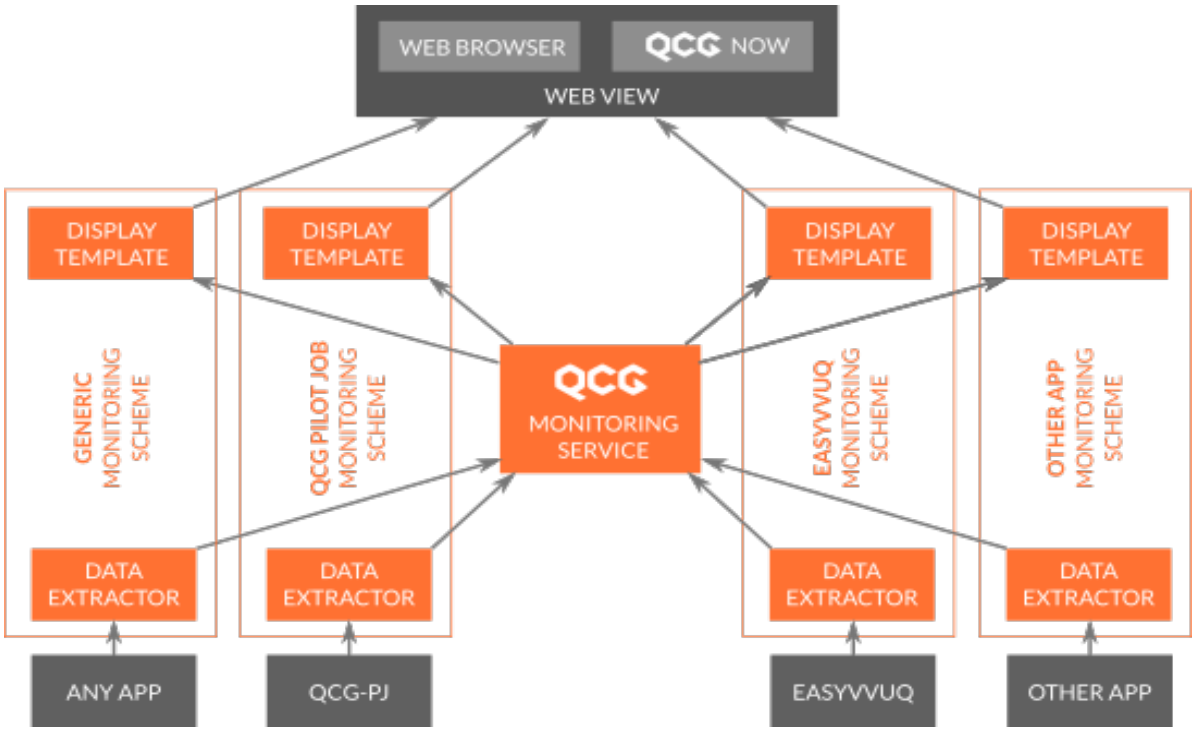

Figure 6 - . QCG-Monitoring system architecture

Since VECMA users are mainly interested in monitoring during an execution of QCG Pilot Job and since the QCG Pilot Job is employed in many places of the VECMA system, we have decided that this VECMA component needs to be monitored. To this end, the specific scheme for monitoring the execution of a QCG Pilot Job has been created. An exemplary monitoring view for the execution of a QCG Pilot Job is presented in Figure 7.

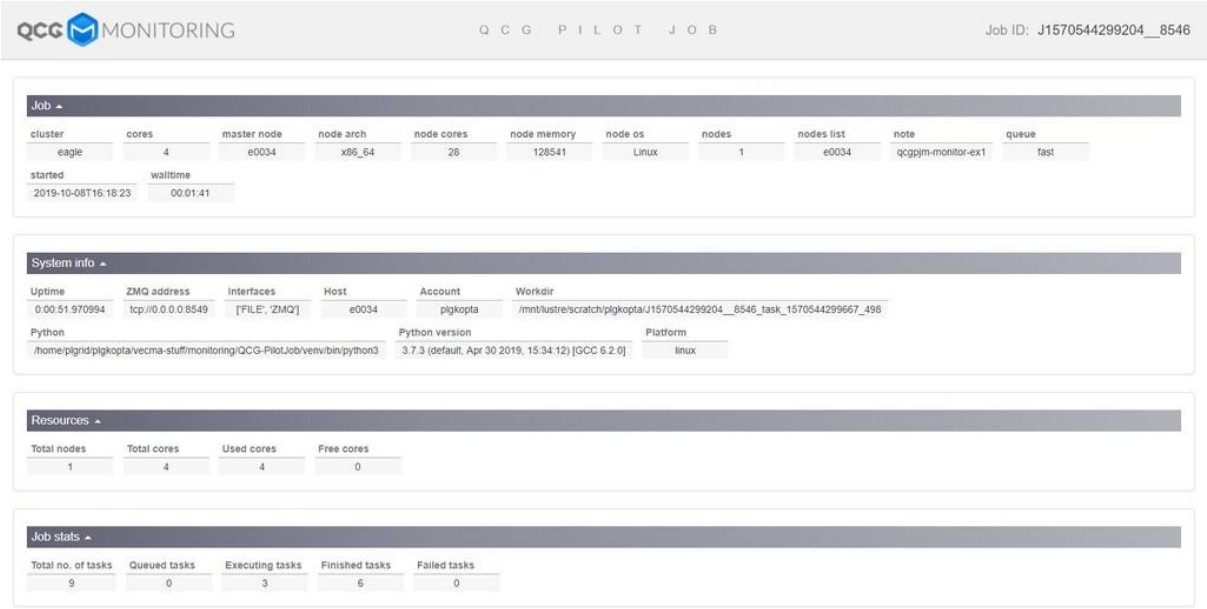

Figure 7 - QCG-Monitoring webpage for tracking progress of QCG Pilot Job

Ongoing work aims at creating a dedicated scheme for monitoring VVUQ workflows executed by EasyVVUQ and QCG Pilot Job. The new scheme will hide technical details related to the execution of QCG Pilot Job behind conceptual terms associated with a typical EasyVVUQ run, making the scheme more practical and easier to use by scientists.

Furthermore, in the last few months, QCG-Monitoring was successfully integrated with the QCG-Now application. Thanks to this integration, all jobs submitted with any QCG tool can be monitored directly from the QCG-Now main window.

Jupyter Notebook

The VECMA toolkit introduces a range of innovative solutions to help in VVUQ of contemporary multiscale applications. Some of them, like for example EasyVVUQ, are specialized, domain oriented programming libraries. These solutions are designed by VECMA consortium and therefore they are seamlessly used by VECMA users. However, although we do our best to provide intuitive and user-friendly interfaces, including well-defined programming APIs, the intrinsic complexity of the VECMA scenarios can be significant for external, especially inexperienced researchers. A particularly unnecessary burden would occur if a new user interested in user-level tools and libraries had to configure the entire programming and execution environment. For that reason, there is a clear need to offer a training platform, where VECMA solutions could be effortlessly and easily learnt and tested. In order to meet such a requirement, as part of WP5 work, the prototype Jupyter Notebooks platform was implemented on PSNC resources. This gaining in popularity technology allows to create and share documents that contain live code, equations, visualizations and narrative text. Every user gets access to a private, isolated and secure web-based interactive computational environment, and therefore it is particularly useful for learning new APIs and command-line tools.

Current work is concentrating on defining of initial VECMA kernel composed of programming and executive environment with pre-configured EasyVVUQ library. The platform can be used in two ways: to host predefined interactive tutorials of VECMA scenarios and to provide easy to access, fully

operational environment in which users code and test their own ideas. As part of further work, the possibility of using the platform to submit and control large-scale calculations using other VECMAtk components (e.g. QCG Pilot job) on the VECMA infrastructure will be verified.

2.2.5 User-level access tools and libraries

From the beginning of VECMA project, and especially during the timeframe of Task 5.5, WP5 organises its activities by following the idea that the actual usefulness of any complex system is highly dependent on the functionality and quality of the tools provided to users. The intention was to deliver tools that will be highly portable and easily deployable. Therefore, we have decided to base access to the VECMA infrastructure on a set of proven generic tools on the assumption they offer good flexibility and potential for adaptation to the specific needs of project rather than develop new tools from scratch.

As a result, by taking into account the requirements and preferences from the VECMA project, and particularly the expectations of user-community, we have decided to provide two interchangeable command-line clients to VECMA, namely QCG-Client and FabSim3, as well as two graphical clients, i.e. QCG-Now and Jupyter Notebook.

QCG-Client, FabSim3 and QCG-Now are already a part of the VECMAtk official release [VECMAtk], and they are continuously developed and improved by partners in WP5 and WP3. The actual schedule for development and integration works is different for each tool. It depends on the usefulness of tool in a specific phase of the project. At month 18, due to the intensive development of VECMAtk and applications, the particular interest is in command-line tools. In order to simplify their installation and deployment on third-party machines, which is a common practice at this stage of the project, these tools have been delivered in a form of docker [Docker] and singularity [Singularity] containers. It is expected that the graphical tools become more important in the forthcoming period, once the VECMA system is increasingly taken up by end users, as well as user-developers.

The following sections describe the motivation behind the selection of each tool and provide information about its current and foreseen usage within the VECMA ecosystem.

QCG-Client

QCG-Client is a basic command-line interface to the computing infrastructure supplied by QCG services, in particular by QCG-Broker. The interface of QCG-Client is created on the basis of popular queuing systems such as PBS or SLURM and therefore it can be easily assimilated by a large number of scientists.

Shortly after the start of the VECMA project QCG-Client was deployed on a dedicated machine in PSNC and integrated with the VECMA's instance of QCG-Broker service. It is currently available for users by logging into qcg.man.poznan.pl via SSH.

Additionally, in order to provide great flexibility to VECMA users, QCG-Client has been containerized and in this form integrated with FabSim3. This way the users may select from both command-line tools, QCG-Client and FabSim3, if they want to access QCG services.

FabSim3

FabSim[1] is a flexible toolkit for automating computational research activities involving large-scale computational resources. FabSim has a strong focus on simplifying and accelerating activities such as creating, executing and analysing models. It can provide a partially automated implementation for an application from a) construction, b) modification, c) instantiation to execution of multiple runs for different policy decisions on the predefined target machine. Currently, various types of remote machines such as Eagle, SuperMUC, Genji, Manny, Cartesius, and ARCHER are supported by FabSim3. FabSim3 stores the target machine-specific configurations in a simple yml file, and applies it to all applications run on the target machine.

In terms of the VECMA infrastructure usage, FabSim3 is integrated with QCG-Client tools at the current stage. By using this feature, VECMA users can easily submit their jobs to the Eagle or any other computing infrastructure using SSH or GSISSH. Additionally, in the case of local installation of QCG-Client tool, job submission by QCG command is also supported. Lastly, FabSim3 is integrated with QCG Pilot Job, allowing FabSim3 users to run ensemble simulations using a one-liner command. The scalability of this integration is still limited, and we are currently revising it to enable an efficient execution of 100-1000s of simulations, given that they have runtimes of at least an hour. This will position FabSim3 as a somewhat less scalable alternative to using QCG Pilot Job directly, but allow FabSim3 users to easily combine Pilot Jobs with all the other elements of their FabSim3 workflows.

As mentioned previously, FabSim3 is also available in both Docker and Singularity containers.

QCG-Now

The ultimate goal of the VECMA project is to deliver not only efficient, but also easy-to-use system for VVUQ of multiscale applications. Since the traditional command-line interfaces by many users may be perceived as too complicated, the consortium decided to put efforts into provisioning of graphical tool that in many cases will be more suitable. The choice fell on the QCG-Now desktop program from the QCG software family that brings basic mechanism for submission and management of computation tasks out-of-the-box.

Within the timeframe of Task 5.5 of WP5, QCG-Now is extended and adjusted to the various VECMA requirements and expectations. The basic version of QCG-Now for VECMA has been initially released as a part of VECMAtk release on month 12th of the project. Soon afterwards, the tool has been significantly altered and several improvements have been applied to enhance its functionality. One of such improvements is the integration with QCG-Monitoring system that results in the possibility to track progress of execution of computing tasks directly from QCG-Now (see Figure 8). This capability is particularly useful for VECMA users who want to keep track of the progress of long-lasting executions of EasyVVUQ and QCG Pilot Job workflows. The other new capability of QCG-Now, which may be useful for practically all users, allows to store frequently used execution schemes as quick templates for their easy reuse in the future.

[1] http://fabsim3.readthedocs.io

Figure 8 - QCG-Now with the embedded QCG-Monitoring view

Jupyter Notebook

The motivation behind usage of Jupyter Notebook has been described in Section 2.2.4. The particular scenarios where Jupyter Notebook will be used are currently evaluated.

2.2.6 VECMAtk M12 Tutorial

The strategic activities of WP5 are supplemented by additional efforts to popularize and advertise of created solutions. One of such efforts was the preparation of a set of tutorials for the month 12 (M12) release of the VECMAtk [VECMA tutorials]. The tutorial titled Uncertainty Quantification in Numerical Models: from simple model on Newton's law of cooling to multiscale fusion model [Fusion tutorial] aimed to demonstrate the benefits of parallel execution of EasyVVUQ with QCG Pilot Job system on local and HPC resources . It was created in close collaboration between WP5 and the Fusion community representing WP4.

The tutorial forms a step-by-step guidance on the usage of several VECMAtk components to perform uncertainty quantification calculations for a large spectrum of potential applications, starting from trivial use-cases to highly complex scenarios. In particular, the tutorial describes the features and capabilities of integration of EasyVVUQ with QCG Pilot Job mechanism in conjunction with QCG-Now and QCG-Client tools.

3 Conclusions and future plans

Realisation of execution environment for the VECMA project is a challenging and still ongoing process. The intensive work of Work Package 5 in the first 18 months allowed us to design and then deliver a fundamental part of the VECMA execution environment. The three layers of the constructed infrastructure, i.e. computing resources, services and user-level tools, provide all means required for the efficient development of VECMA toolkit and VECMA applications in the next few months of the project. However, in order to follow the needs of VECMA in the long run, WP5 is obligated to improve the quality and functionality of the infrastructure in several fields.

First, although the hardware resources as well as core execution services are configured and in place, further efforts are needed to continue the development of QCG Pilot Job system. The system needs to be made more reliable and efficient, but ultimately it should also be extended e.g. with a common queue functionality allowing highly-demanding computational scenarios to execute on many clusters. Also the existing minimalistic integration between QCG Pilot Job and EasyVVUQ materialised in the EasyVVUQ-QCGPJ wrapper should be transformed into a more extensible and user-friendly API.

In addition, in the forthcoming part of the project, the focus will be placed on the further development of user-level tools and peripheral components that can enhance practical usability of the system. It is expected that in order to support VECMA end-users, the set of currently-released VECMAtk access-tools in the VECMA system will be adjusted to specific needs of the project and the particular tools will be improved. The VECMA system will be further enhanced with QCG-Monitoring service that allows one to easily track the execution of EasyVVUQ and QCG Pilot Job workflows and possibly also individual applications. The additional aim of WP5 in this context is to instantiate the Jupyter Notebook technology and make it available to the broader VECMA community.

4 References

1. https://www.vecma.eu/wp-content/uploads/2019/03/VECMA_D4.1_ApplicationReadinessReport_MPG_v1.1_20190308.pdf
2. https://www.vecma.eu/wp-content/uploads/2019/03/VECMA_D5.1_Architecture_PSNC_v1.0_20190312.pdf
3. https://wiki.vecma.eu/vecma-wp5
4. https://wiki.vecma.eu/access_eagle
5. MAPPER project: https://cordis.europa.eu/project/rcn/95304/factsheet/en
6. ComPat project: https://cordis.europa.eu/project/rcn/197534/factsheet/en
7. QCG webpage: www.qoscosgrid.org
8. QCG Pilot Job: https://github.com/vecma-project/QCG-PilotJob
9. VECMA toolkit: https://www.vecma-toolkit.eu/
10. VECMA tutorials: https://www.vecma-toolkit.eu/tutorials/
11. Fusion tutorial: https://github.com/vecma-project/VECMAtk/blob/master/tutorials/M12/easyvvuq-qcgpj/guide/EasyVVUQ%20%26%20QCGPJ%20-%20Tutorial.md
12. Docker webpage: https://www.docker.com
13. Singularity webpage: https://sylabs.io/